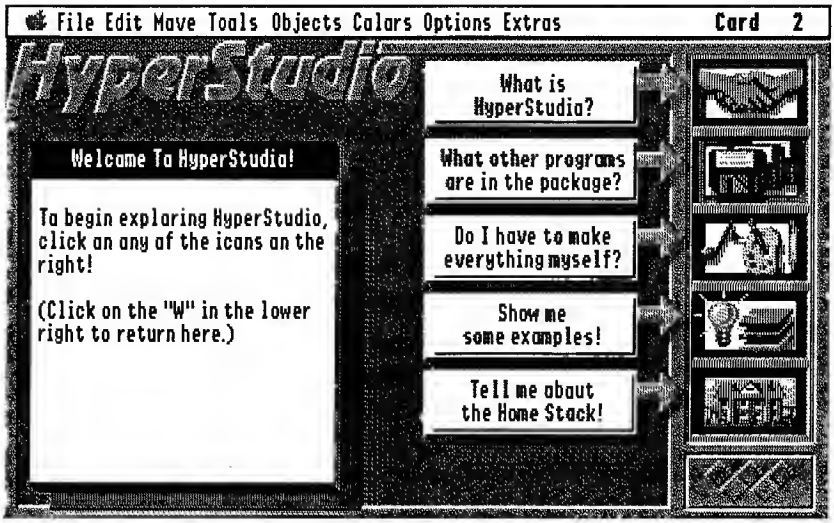
HyperStudio®

reference

HyperStudio comes complete with...

- Award-winning HyperStudio Software
- Sound Digitizing Hardware
- Microphone
- Clip-Art and Clip-Sounds
- Sample Stacks



TM

Roger Wagner™
PUBLISHING, INC.

HyperStudio®
Reference Manual

the premier hypermedia/
multimedia system for
the Apple IIGS

by Michael O'Keefe, Dave Klimas, and Jeff Smith

(619) 442-0522

Design and Interface: Roger Wagner and Michael O'Keefe
Programmers: Michael O'Keefe, Dave Klimas, and Jeff Smith
Contributing Programmers: Roy Bannon, Garrick Toubassi, Ken Kashmarek, Steve Allen, Mike Nuzzi, Terry Romine, and Eric Mueller
Contributing Artists: Bob Cavey, Bo Monroe, Donald McIntosh, and Greg Roach
Contributing Stack Designers: Bo Monroe, Steve Allen, Donald McIntosh, Greg Roach, and Dean Esmay
HyperStudio Reference: Eric Mueller, Bert Klimas, and Roger Wagner
HyperStudio Tutorial: Della Smith and Eric Mueller
Special Thanks: Garland Buckingham, Valerie Laird, Leslie O'Brien, Sandy Goez, and Debbie Midgley
Dedicated To: Pam Wagner, Bert Klimas, Andrea Marcucci, and Della Smith

Produced by:
Roger Wagner Publishing, Inc.
1050 Pioneer Way, Suite P
El Cajon, California 92020

Customer Service and Technical Support
(619) 442-0522
9:30am - 5:30pm Pacific time
please have your HyperStudio serial number ready

ISBN 0-927796-39-2

revision 1.1 12/13/91 and 1/2/92; first printing, January 1992

1 3 5 7 9 10 8 6 4 2

TABLE OF CONTENTS

chapter

1

Introduction

This section introduces you to the multimedia program for multi-talented minds, HyperStudio.

Introduction to HyperStudio

Welcome to the world of HyperStudio and hypermedia! HyperStudio provides you with the complete system to create things on your computer that will make it more useful, more educational, and just plain more fun than you can probably imagine. HyperStudio is an application that allows you to create cards containing text, color images and sounds. Buttons on the cards allow you to navigate through the stack of cards—and beyond—in whatever order you choose.

What is Hypermedia?

If you're new to the subject, it's probably appropriate to answer the question, "What is hypermedia?" You're already familiar with the various ways of exchanging information that currently exist. Newspapers, books, television, radio, movies—all these are the media. They are the channels that handle the mass distribution of information. This can be the printed word, a colorful graphic, or an electronic image. For the most part, the current ways of conveying information in the modern world are passive: you act as a silent viewer of the information presented.

The word "hyper" is derived from a Greek word meaning extreme or beyond. The word hypermedia was coined to describe communication beyond the traditional media.

Imagine starting with something common, like a book. This is a typical form that information takes in our world. Imagine seeing not only text and pictures on each page of the book, but certain areas that you could touch with your finger that would suddenly expand into a new page of information, or perhaps take you to an entirely different part of the book... perhaps even to a completely different book. While you're thinking of that, wouldn't it be nice if the pictures on the page could even be animated to show a bird flying, or the path that an army followed during the Civil War? Better still, imagine hearing the sounds of the battle while you read the text!

Hypermedia paves the way to exploration, communication, and creativity. It's a very expressive environment because of the easy way that different forms of media can be mixed. If you were doing a report on Bach, you could write a report talking about his life—and with hypermedia, easily add on a sound bite of Bach's music as you perform it on a keyboard, and put in some digitized photographs of Bach.

This is exactly what hypermedia offers, and what HyperStudio delivers! With HyperStudio, you can create entirely new hypermedia applications ("stacks"), or work with existing stacks and modify them to your liking.

What HyperStudio Can Do For You

At this point, the relevance of hypermedia to your situation may not be obvious, so let's look at some specific ways that HyperStudio may be of immediate use to you.

When you made your decision to buy the Apple IIGS, you were probably attracted to its terrific graphics, realistic sound abilities, and overall system power. The Apple IIGS is also one of the easiest-to-use computers ever created.

You've probably seen individual programs that take advantage of a few of these features on the Apple IIGS, but found it difficult to find something that combines them

all in one environment. HyperStudio's first big feature is that it gives you one environment where you can use all the abilities of the Apple IIGS in a variety of ways.

This brings us to HyperStudio's second big feature: it makes it easy to create your own software projects on the computer, without having to spend much time, or needing a lot of technical knowledge.

If your family has young children learning the alphabet, the names of states, or other skills, you could try to buy a program that teaches these ideas. Chances are, though, that it would be somewhat expensive, take time to find, and might not be exactly what you want.

With HyperStudio, it's easy to create a few screens with letters of the alphabet, or words and common objects, even a map of the United States, and add your voice prompting the user to click the right image or state. The whole family can participate in creating this kind of software, helping with voice prompts and responses, creating graphics, recording sound effects, and so on.

However, HyperStudio isn't just for creating software for young children. What it is best at is communicating ideas and information. If you're a teacher, this can be lessons, presentations, even tests on many kinds of material. From foreign languages to complex concepts, HyperStudio is the perfect tool, and requires no more effort than preparing any other kind of lesson presentation.

If you're a student, HyperStudio offers the most dynamic environment for reports ever created. You'll find reports can even be put on videotape to share with people who may not have a computer. HyperStudio can also be used to record and organize notes and ideas for the more traditional written report.

Even if you are neither teacher nor student, but would like to share knowledge you have with others, record your family tree, introduce your business customers to new products or show them how to find something they're looking for, keep track of your music or magazine collection, or any number of other applications that hypermedia is an ideal solution for, then HyperStudio is for you.

The HyperStudio disks come with many example stacks to get you started, and give you ideas for your own stacks.

This manual will cover how to install the HyperStudio software and sound card, and provides a detailed reference to all of the wonderful functions and options of the software. Once you've installed the hardware, we suggest you go through the HyperStudio Tutorial to get started.

Then, when you're familiar with the general operation of the program, you should casually browse through this reference manual to see what other features are available. However, you needn't try to memorize each function. The best approach is to use this manual as a "toolbox," or "encyclopedia of options," and return to it on those occasions when you're looking for "the next step" in a given project. We think you'll be pleasantly surprised to discover that HyperStudio is the most elegantly designed hypermedia authoring system available on *any* computer. We trust you'll find it to be a major part of the continued enjoyment of your computer.

Conventions Used In This Manual

This manual uses several visual cues to help you use it better. They are:

❖ **Shortcuts and Tips**

This icon appears next to notes that will let you work faster and easier with HyperStudio.

✔ **Advanced User**

This icon appears next to notes that apply to HyperStudio when the "Advanced User" checkbox is marked in the Preferences dialog box.

☞ **Novice User**

This icon appears next to notes that apply to HyperStudio when the "Advanced User" checkbox is *not* marked in the Preferences dialog box.

Support and Help

As you're using HyperStudio, you may run into problems that this documentation doesn't cover. In those cases, please call our technical support hotline, at:

(619) 442-0522

9:30am – 5:30pm Pacific
Monday through Friday

Please have your serial number ready.

We'll be happy to help you!

chapter

2

Setting Up

This section will help you move HyperStudio from its box to your computer. All you will need is a flat-bladed screwdriver and six blank disks.

System Requirements

HyperStudio requires an Apple IIGS with at least one megabyte of total RAM, and a 3.5" disk drive. It is completely compatible with GS/OS, and can be run on hard disks as well as networks such as AppleShare, Corvus, Digicard, and the Josten's Integrated Learning System.

We recommend you set your RAM disk to 0K (zero K). Use the RAM disk option in the Control Panel. This will give HyperStudio access to all the RAM in your computer for the stacks.

The memory requirements for a stack depend on what is in the stack. Different backgrounds, graphic objects, and embedded sounds add to memory needs. With no embedded sounds, and simple card backgrounds, a stack typically holds about 20 cards, although this number varies depending on your stack design. HyperStudio optimizes memory use by compressing memory, and other techniques.

Digitized sound requires approximately 4K of RAM per second of sound. (A ten second sound requires 40K of memory.) However, HyperStudio does not require that every sound in a stack be loaded at once. The amount of sound available in a stack is more a function of disk size.

Many of our customers run HyperStudio on a one megabyte, single-drive Apple IIGS without any problems.

HyperStudio keeps track of your remaining free memory as you work on a stack. It also keeps a "reserve" memory area. As your stack approaches the maximum size possible, HyperStudio will open a window, telling you it is getting low on memory, and suggesting that you save your stack and re-start HyperStudio. Unless you are attempting to add something very large (such as a long recorded sound, for example), you should be able to complete what you're doing, save the additions, and then quit HyperStudio. At that point, you can re-start HyperStudio, re-load your stack, and continue.

(A word of explanation: memory in the computer is not just a huge empty area. It is actually broken up into "chunks," and as you add clip-art, sounds, edit buttons, and so on, the chunks of memory may get too small to perform certain operations. Re-starting the HyperStudio program does some "housecleaning" that re-orders memory into a more usable arrangement. This happens with all computers and programs, but is just more noticable when your stack reaches the upper size limits. If you encounter this often enough to inconvenience you, try keeping your stacks a little smaller. You can always check the stack size, and the amount of memory remaining, by choosing "Stack Info" from the Objects menu.)

What's Included

Your HyperStudio package contains:

- six diskettes
- a sound digitizing board with mounting hardware
- a microphone and stand
- two manuals—this reference and a tutorial.

The six disks included are non-copy-protected, and may be installed on a hard disk. The disk names and contents are:

`/HyperStudio`	contains the HyperStudio program itself and the Home Stack.
`/HS.System`	the system boot disk. It contains the run-time version of HyperStudio, and is also a complete "user disk."
`/HS.Art`	holds the clip-art included with the HyperStudio package, which you can use in your own stacks.
`/HS.Sounds`	contains "clip-sounds" for use with HyperStudio, Sound Shop, and Sight 'n' Sound.
`/Samples.1` and `/Samples.2`	two disks full of sample HyperStudio stacks to show you a little bit of what others are doing with HyperStudio, and to give you ideas for your own stacks.

Beginning Installation

Before starting installation, you should perform two important steps:

Backing Up Your HyperStudio Disks

Before doing anything else, please back up your HyperStudio disks and put the originals in a safe place. The chance of physically damaging a disk in normal use is small, but it's still a good idea to make a duplicate copy, or "backup," of the original HyperStudio disks. This process is very easy and takes only a few minutes. It is a worthwhile provision for the possibility that something ever happens to the disks you use regularly. (Replacement HyperStudio disks are available from Roger Wagner Publishing for $10, but if you back the disks up yourself, you can always return to your originals should something happen to your backup copies.)

Take a moment now to make backup copies of the HyperStudio disks. There is not much space remaining on the HyperStudio disks for additional files, so you should also make a data disk for new stacks.

If you are not sure how to make backup copies, or create data disks, see appendix C, "Copying a Disk."

Read the `Read.Me` file

After you've backed up the HyperStudio disks, boot the `HS.System` disk and examine the "Read.Me" file, which will be presented to you on the first screen. It contains any last-minute additions to changes to HyperStudio, as well as any corrections to this documentation.

Installing the Hardware

Part of the HyperStudio package is a set of sound digitizing (recording) equipment. These tools make it easy to record or edit your voice, music, and sound effects for your stacks.

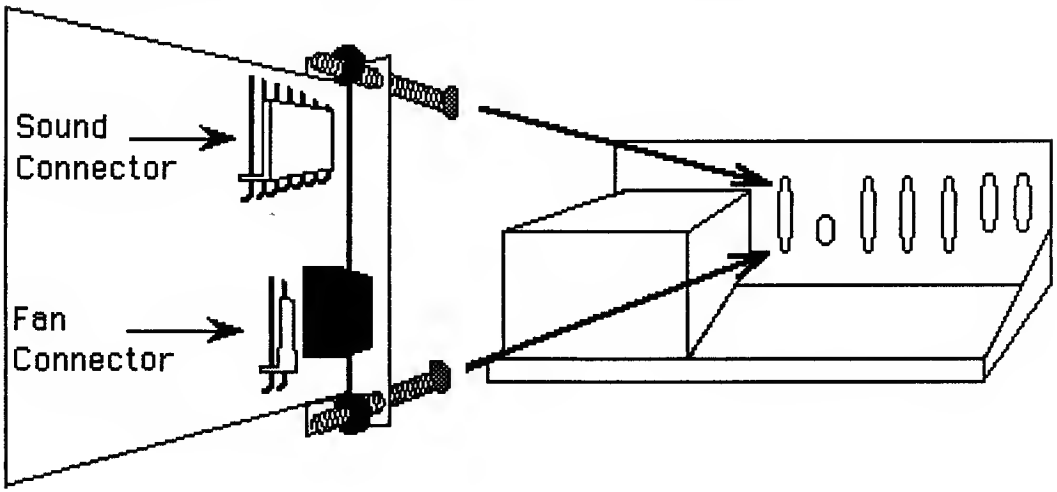
The most important part of these tools is the digitizing board, itself. Verify that the following items are in the box:

- one HyperStudio digitizing board
- two bolts and two nuts
- one microphone with stand

You also need a small flat-bladed screwdriver.

1. Turn off the computer, but don't unplug it from the wall outlet.
2. With the palms of your hands on the sides of the back of the lid, use your fingers to gently push the release tabs on the back of the computer toward you. While pressing the tabs, use the palms of your hands to lift the back of the cover up and off.
3. Look down at the computer. With the front closest to you, you'll see seven "slots" toward the back of the computer. Another one is more to the right and closer to the front.

   The digitizer doesn't go in any of these slots! This has two immediate benefits: first, your recordings will have much less background noise than recordings done with boards that must go in a slot. (The computer is the source of background "hiss" or occasional "pops." This electronic noise is strongest in the slot connectors.) The second benefit of not placing the digitizer board in a slot is simply that the number of slots in your computer is limited, and the HyperStudio digitizer board won't use up any of your valuable "real estate."



4. Look toward the back of the computer. You will see openings ("cutouts") in the back panel that are closed off with plastic covers. On the inside of the back panel, each cover is attached with a metal clip. Identify the opening at the far left (closest to the power supply). Turn the clip counter-clockwise one-half turn. The clip should now be easy to remove. Gently pull it directly toward the front of the computer. The plastic cover will also fall free.
5. It is behind this opening, *inside* the computer, that you will mount the circuit board. Partially thread a nut onto each bolt, both top and bottom, through the holes on the digitizer's mounting bracket. Leave as much free play in the bolts as possible.
6. Remove any interface card in slot one (the slot closest to the power supply) temporarily, to give yourself room to work in.
7. The power for the digitizing card is picked up from two small wire posts (marked "fan") near the back of the computer. These posts are usually used for powering an internal fan in the computer. If you have an internal fan, disconnect it (for now) from the power connector.
8. Locate the small two-wire connector coming from the digitizing card. Push it onto the post marked "fan" near the back of the computer. The plug is "keyed," which means that it is difficult to install backward. Just to be sure you've positioned it correctly, verify that the red wire is on your left as you look toward the back of the computer (with the front of the computer closest to you).
9. If you have an internal fan, attach its connector to the two-pin connector on the digitizing board. This replaces the function of the connector on the computer's main circuit board. It can be used for anything that would otherwise require the connector used by the HyperStudio digitizer card.
10. Insert both bolts into the digitizer board's mounting bracket.
11. Lower the head of the lower bolt into its slot in the cutout. Use the tip of your finger, or the screwdriver, to push the head of the top bolt down as you pass it through the cutout. Once the heads of both bolts are through the cutout, adjust them into the proper position in each slot.

Alternate method

If you are having problems installing the digitizer board in the computer, you may wish to try this alternate installation method.

Insert only one bolt through the lower hole in the circuit board's mounting bracket. Insert it so the head is on the side of the plate opposite the circuit board, and the threaded end points toward the board. Now attach the nut to the bolt. Do not turn the nut more than 3-4 turns. You want a lot of open area left between the nut and the head of the bolt.

With the lower bolt and nut as loose as they can be without the nut falling off, pull the head of the bolt away from the bracket. With the 7-pin connecter on the right side of the board (facing up) place the board and its mounting bracket against the inside back of the computer so the loose bolt drops into the notch at the bottom of the cutout. The idea

is to fit the card into position with the head of the bolt on the outside of the back of the computer.

Once in position, gently tighten the bolt and nut with your fingers. Do not tighten them completely! You want to leave a little "slack" so you can insert and tighten the upper bolt. You may have to gently push down on, or fold to the right, the wires that go from the power supply to the computer's main circuit board to make room for the digitizer card.

Note: the first opening (from the left) in the back of the computer is generally the most convenient location for the HyperStudio digitizer. You can use any of the other cutouts on the back of the computer to mount the digitizing board except the small circular cutout, but they will probably require that the slot in front of the cutout does not have an interface card in it. You are welcome to use whatever position is the most convenient for you.

Insert the bolt through the top hole in the back of the computer and the mounting bracket on the card. Attach the nut to the end of the bolt on the inside. It is a little cramped here. With patience you should get it in place. It works best if you turn the head of the bolt with the screwdriver, rather than the nut. If you drop the nut while trying to attach it to the bolt, don't panic. You can lift out the digitizer board, retrieve the nut, then replace the board and try again. (If your screwdriver is magnetized, just touch the end of the screwdriver to the nut to pick it up.)

12. Tighten both the top and bottom bolts once the board is in place. *Do not overtighten!* It is not necessary to turn the bolt head until it won't budge. Just make sure that the connecting bolts are firm.
13. Replace any other cards you removed from the slots.
14. Find the 7-pin connector marked "J25." It is toward the front of the computer, on the right side, next to the speaker connector. If you have another sound-related hardware device in your computer, such as the Applied Engineering Sonic Blaster or Audio Animator, unplug that connector from this post.
15. Plug the long ribbon cable from the HyperStudio digitizer into the 7-pin connector on the computer's main circuit board. It is also "keyed" so that connecting it wrong is very difficult. However, you should verify that the red wire on the ribbon cable is closest to the front of the computer as it enters the connector at J25. The cable is deliberately made just long enough to reach the J25 connector—this helps reduce electrical noise. If you have a lot of cards in the slots of you computer, run the cable over the tops of the cards. Don't try to snake the cable between cards.
16. If you have disconnected an existing sound hardware device, you can now plug it into the replacement connector at the top of the HyperStudio circuit board.
17. The installation of the board is done. Take one last look inside before putting the cover back on the computer. Make sure everything is properly attached, and that there are no loose wires left hanging. Now you can put the cover back onto the computer.

18. Plug the microphone into the connector on the mounting plate for the HyperStudio digitizer. Be certain that you have the microphone plugged completely into the jack. There should be an audible click when the microphone plugs in completely, and the black plastic should be completely flush with the digitizer board, with no part of the microphone's metal plug visible.

That's it! The HyperStudio hardware is installed. If you need to move your computer, just unplug the microphone from the digitizing board. It should never be necessary to remove the circuit board. (Check the tightness of the connecting bolts occasionally if you move your computer often.)

Automatic microphone switching

The microphone connector on the HyperStudio digitizing card has a special convenience feature: a switching function is built into it.

When the microphone is plugged in:	When the microphone is unplugged:
The HyperStudio digitizer is active. Any alternate sound input device connected to the card is disabled.	Any sound input device is enabled. It can be used in the customary way.

This makes it easy to switch between the HyperStudio digitizer and another device.

The sound output function of any device (stereo card, etc.) connected to the HyperStudio digitizer is always active whether or not the microphone is plugged in.

Installing the Software

Now that you have the hardware in place, it's time to install the software. The first step is to be certain that you have the software backed up and that you are *not* using your original disks. Next, you need to check the computer's Control Panel settings, as discussed in the "Control Panel Settings" section, below. Then, if you will be using HyperStudio on a floppy-disk system, please read "Using HyperStudio With Floppy Drives." Otherwise, you may skip to "Installing HyperStudio on a Hard Disk or Network."

Control Panel settings

Before using HyperStudio, you need to verify several settings in your computer's Control Panel. The Control Panel is where many aspects of the system (such as the machine's speed and sound volume level) are controlled.

RAM Disk

On an older Apple IIGS machine (ROM 01), the first two selections will be "Minimum RAM Disk Size" and "Maximum RAM Disk Size". Use the left and right arrow keys to set both values to ØK, if necessary. Note: although different values are allowed in these two settings, experience has shown that the system operates better when both values are the same. For HyperStudio on a 1.25Mb machine, the RAM disk setting should be at zero to allow maximum memory for your stacks.

On the newer Apple IIGS machines (ROM 03), the first selection will be "Select RAM Disk Size." Use the left and right arrow keys to set this value to ØK, if necessary.

Sound

The "Volume" setting should be at least at the standard (default) level, or higher. (When the volume is at the default level, a checkbox will appear to the left of the word "Volume".) The real test is to press the space bar: if you can hear the system beep reasonably well, then that's fine. You can increase the sound level with the right arrow key.

System Speed

The System Speed should be on "Fast." Sometimes Apple IIGS machines are set to the "Normal" (slower) setting to accommodate older Apple II programs, particularly those that use a game controller or joystick.

Using HyperStudio With Floppy Drives

To use HyperStudio with a floppy-based system, simply boot the `/HS.System` disk. (Remember to *always* use backup disks, *never* your originals!) After the system boots, you will be prompted to insert the `/HyperStudio` disk. If you have two disk drives, you may want to insert it into the second drive; otherwise, simply eject the `/HS.System` disk and insert the `/HyperStudio` disk.

Note that, on a one drive system, you will occasionally be prompted for the `/HS.System` disk. Simply follow the directions on the screen whenever the computer asks for a disk.

Installing HyperStudio on a Hard Disk or Network

To install HyperStudio on a hard drive, you need to be certain that your hard drive has System Software 5.0.4 or later. You can get the latest version of the Apple IIGS System Software from your local dealer or user group for a minimal charge (usually for free).

Boot your hard drive as normal, and from your program selector (which may be the Finder, Salvation: Wings, or ProSel 16), launch the Installer program, found on the `/HyperStudio` disk. When the Installer is loaded, you will have several options on the left side of the screen, to install HyperStudio on your hard drive. They are:

`All files on hard drive [or network]`	This option installs the entire HyperStudio package on your hard drive in a folder called "HyperStudio" on the volume you specify. If you have the network version of HyperStudio, this will allow you to install it on a network, as well. *Important note:* this option installs HyperStudio, the home stack, and all of the sample stacks, accessory programs, clip-art, and clip-sounds included with the package. It is essentially a combination of all of the installer options listed below. After selecting this installation option, you do not need to use any of the options below.
`Small installation`	By selecting this option, you can install only the HyperStudio program and the home stack in the folder "HyperStudio" on the volume you specify. This is useful if you have limited space on your hard drive or simply don't want the entire package.
`All sample stacks`	Selecting this option installs the sample stacks included with HyperStudio. They will be placed in the folder "HyperStudio" on the volume you specify.

`All accessory programs`	By selecting this option, you can install the four HyperStudio accessory programs: Sound Shop, SNS.320, SNS.640 and Sound Browser. They will be placed in the folder "HyperStudio" on the volume you specify. Note that the HyperStudio standard home stack will automatically know how to find the accessory programs. (This is assuming that you have HyperStudio and the standard home stack is in the same folder.)
`All clip-art`	Selecting this option will install all of the clip-art on the `/HS.Art` disk in a folder called "HS.Art" in the "HyperStudio" folder on the disk you specify.
`All clip-sounds`	This option will install all of the clip-sounds on the `/HS.Sounds` disk in the folder "HS.Sounds" in the "HyperStudio" folder on the disk you specify.
`ACE toolset`	This option is only necessary if you are using a hard drive. It will copy the "ACE toolset" from your Apple IIGS System Disk to your hard drive. (The ACE toolset is used by HyperStudio but isn't a part of the normal system installation procedure.) You only need to select this option if you are installing HyperStudio on a hard drive.

Creating User Disks for Others

Although the HyperStudio program itself cannot be given to others, you can give out copies of a miniature version of HyperStudio (called the "run-time version") that has no editing functions. This program is called `HS.Sys16` can be found on the `/HS.System` disk.

Bootable User Disks

The easiest way to make a bootable user disk for someone else is to make a copy of the `/HS.System` disk, and then replace the `Home.Stack` file with your own stack. (If you name it `Home.Stack`, it will automatically be loaded when the disk is booted; otherwise, the user will be prompted to find your stack.) For information on copying a disk, see appendix C.

Please be sure to read "License Considerations for `HS.Sys16`," below.

Non-bootable User Disks

Sometimes, there is not enough room for your stack to fit on a bootable user disk. In these cases, you can simply place your stack (named `Home.Stack`) on a data disk along with the `HS.Sys16` file (the run-time version of HyperStudio). The disk will not be bootable, but the end user can boot their Apple IIGS System Disk and then launch the `HS.Sys16` file.

Please be sure to read "License Considerations for `HS.Sys16`," below.

Note that, if the eventual user of your stack will own HyperStudio, there is no need for the `HS.Sys16` file—they can simply load your stack with their copy of HyperStudio.

License Considerations for `HS.Sys16`

There is a small license fee for `HS.Sys16` if you will be distributing it for commercial purposes. The license fee is $100 per year per product, or $100 per year for HyperStudio-based publications. (This way, a magazine that publishes a monthly disk which includes the `HS.Sys16` file wouldn't have to pay $1200 per year.)

Note that shareware and freeware are *not* considered commercial purposes, as well as user groups creating a disk-of-the-month (DOM). In these cases, `HS.Sys16` may be distributed at no charge.

If you have any questions, are wondering if your purpose is commercial or not, or just want to get a copy of the `HS.Sys16` license agreement, please contact Roger Wagner Publishing.

More about `HS.Sys16` and Languages

While sounds, NBAs, and transitions are all stored with a stack, languages are stored within HyperStudio itself—or, in this case, the run-time versions of languages are stored within the run-time version of HyperStudio.

The `HS.Sys16` file comes with the run-time version of SimpleScript built in. If you design a stack that does not use SimpleScript, at some point in the future, it may be

possible to make the run-time version of HyperStudio smaller by simply removing the run-time version of SimpleScript. (The resulting `HS.Sys16` file not be able to properly run a stack that uses SimpleScript, but if you're simply pairing it with a stack of your own design and you know that it will not be used with any other stack, this should not be an issue.)

Currently, management of languages, including SimpleScript and any third-party languages, will be handled by utilities that others will provide. HyperStudio doesn't automatically add or delete languages from the `HS.Sys16` file.

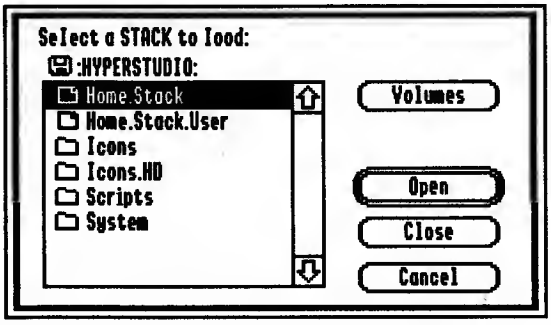
File Location Dialog Boxes

When HyperStudio cannot locate a file, a dialog box will ask you to insert the needed disk. If you cancel that dialog box, HyperStudio will then ask you to locate the file on any disk or folder with the standard file dialog. If you cancel the standard file dialog, HyperStudio will skip the item in question when building a card or executing a button action, and will continue.

For example, if HyperStudio tries to play a disk-based sound attached to a button and can't find the sound file, it will first ask you to insert the disk where it thinks the sound is located. If you choose to click "Cancel" instead of inserting the requested disk, or if the disk is already online (and the file can't be located), HyperStudio will then prompt you with a standard file dialog and ask you to find the sound file "manually." Clicking "Cancel" at this point (instead of specifying a sound file) will cause HyperStudio to simply not play any sound.

The Standard File Dialog Box

The standard file dialog box, shown to the right, appears whenever you open a disk file. (For example, the standard file dialog box appears whenever you select "Open Stack...," "Add Clip-Art...," or any other option that requires a file on the disk.)



To use the standard file dialog box to select a file, first highlight the file in the list by clicking its name, then click "Open." (You can also simply double-click the filename.) To open a folder, do the same thing: highlight the name of the folder, then click "Open," or simply double-click the name of the folder.

To close a folder (and "back up" a level in the directory hierarchy), click the "Close" button or click the current folder or disk name, above the list of filenames. If you are already at the root directory (that is, if you have no folders open), clicking the disk name will display a list of all online volumes.

Clicking the "Volume" button will also display a list of all online volumes, allowing you to select one, and clicking the "Cancel" button will indicate that you wish to cancel the operation.

Last-used Directories

When adding sounds or graphics, HyperStudio remembers the most recently used directory for various different *kinds* of data. This way, you may store all of your graphics in one directory (for example), and the first time to go to add graphics (via any of the commands that add graphics: "Load Background...", "Add Clip-Art...", "Add a Graphic...," and the command to load an icon to be attached to a button), you can load art from your graphics directory. The next time you use any of those graphics commands, HyperStudio will automatically remember your graphics directory and take you there. If you have many HyperStudio-related files, you may find that organizing folders into different kinds of data is both useful and necessary.

HyperStudio remembers six directories: the sounds directory (used when loading or saving a sound from or to disk), the text directory (for bringing in text from disk when creating a text item), the link-to-stack directory (used when a button action links to another stack), the graphics directory (used by "Load Background...", "Add Clip-Art...", "Add a Graphic...," and the command to load an icon to be attached to a button), the stack directory (used by "Open stack..." and "Save Stack As..."), an application directory (used when a button action connects to an application) and the 'add-on' directory (used for loading transitions and NBAs).

All About Home Stacks

The home stack is the central stack which HyperStudio navigates from. It's the stack that HyperStudio tries to load when it starts and the stack that all other stacks should eventually return to.

The default home stack (which is also installed on your hard drive during the normal HyperStudio installation procedure) is more informational than navigational. It tells you a little bit about hypermedia and HyperStudio rather than helping you locate other stacks on your disk.

As a result, you may want to replace the provided home stack with one that suits your needs better. To do this, simply name your stack `Home.Stack` and place it in the same directory as HyperStudio. It will automatically be loaded on startup and by pressing ⌘-H, the user will be returned to the `Home.Stack`.

If you receive a stack set from a friend that includes a stack called `Home.Stack`, you want to be careful to not replace your usual `Home.Stack` with the new file. In most cases, you will want to create a new folder when you receive a stack of stacks that all work together and have their own home stack.

For example, let's say you've received a disk containing a French Poodle Information stack. Examining the directory shows four HyperStudio stacks: `Home.Stack`, `Grooming`, `Feeding`, and `Attitude`. You know that these four stacks work together and through experimentation (or by reading the disk label), you discover that starting the French Poodle Information stack is as simple as opening the file `Home.Stack`.

One of the easiest ways to install this on a hard drive is to make a folder in your HyperStudio folder called `FPI.Stack`, and copy all of the files from the floppy disk into the new folder. You could then make a button on your Home.Stack that connects to the stack called `Home.Stack` *inside of the `FPI.Stack` folder.*

To return from the French Poodle Information stack to your original home stack (which resides in the same directory as the HyperStudio application), you need to make a button in the French Poodle home stack that connects back to your own home stack. You can't simply make a button with a button action of "Connect to home stack" because the most recently loaded stack named `Home.Stack` was the French Poodle Information home stack—and as a result, the button marked "Connect to home stack" won't go anyplace.

There are two ways to avoid all of this. The first is to design your multi-stack sets so that the "main" stack (which accesses the others) is called *topic*`.Home`. So, for the example above, ideally, the stack that began the French Poodle Information set should have been called `Poodle.Home`. This would have avoided having to place a special button in the `Poodle.Home` file to return to your original home stack—a simple "Connect to home stack" would have gone to the main HyperStudio home stack (since it was the last stack loaded that was named `Home.Stack`).

The second solution is to find the button in the French Poodle Information `Home.Stack` and change it from "Connect to home stack" to "Connect to another stack...", then link it to your regular (main) HyperStudio `Home.Stack`.

Data Access Methods

HyperStudio can store information (such as graphic objects and text items) in two different ways: embedded and disk-based. If you are an advanced user, you can select what access method you'd like. As a novice user, all data is stored in stack, embedded. Both options are explained in detail, below.

- *Embedded Data:* When you add a graphic or text item to a card, or a button containing a sound, the default is to store the data for that item within the stack itself. This is convenient and gives faster movement between cards in a stack. The other advantage of this method is that the stack is very transportable—since everything that makes up that stack is within the file itself, moving the stack from disk to disk is as simple as copying a single file. This type of object is also called an *embedded data* object.

  However, suppose you want to create a stack that uses literally hundreds of graphics, text, and sound items. Even with megabytes of memory, you would soon want to have more data in your stack than could fit in the computer's memory all at once. It is also possible that someday you may want to access data, such as that on a compact disc or laser videodisc, that *cannot* be physically embedded in your stack. In those cases, HyperStudio provides an alternate way of accessing text, graphic, and sound data.

- *Disk-Based Data:* Disk-based access allows you to leave the object file in its location on disk, instead of keeping it in memory with the stack. HyperStudio will go back to the disk whenever it needs the data. (An object setup like this is also called an *extended data* object.) Although this gives a slower transition time between cards, it decreases the memory required for your stack, and increases the overall size capability of your stack and the amount of data it can hold.

  Another advantage of the extended data type is that other users of the system can update the data displayed on a card independent of HyperStudio. For example, you may have a stack that orients new employees to your company. If you kept your organizational chart in an AppleWorks file and referenced it in the stack as an extended data object, changing the chart in the stack would be as simple as changing the word processing file.

  Extended data makes instant distribution of all kinds of information possible. This could include memos to staff, names in an organizational chart, instructions to students, or often-changing data such as a daily bulletin or stock quotes.

  Here's some suggestions to help you decide which data access method to use when designing a stack:

  The embedded data type is appropriate when the entire stack *can* fit in memory all at once. Stacks done this way are much easier to distribute, and they run faster, as well. The disadvantage is that, as they start to grow, you may need to start making data disk-based.

  The disk-based data type is recommended if the stack will be running in a *stable* environment, the referenced data does not change location, and you want to have common files that are accessed by a variety of stacks. It's particularly useful in a network environment.

  The default in HyperStudio, when shipped, is the embedded data system. This means the user has less to worry about. Stacks are easy to move and operate. However, as an advanced user, you may take advantage of the alternate, disk-based data access method.

Connecting a Laserdisc Player

HyperStudio can control a Pioneer 2200, 4200, or 8000 LaserDisc player attached to the Apple IIGS. Additionally, HyperStudio can control an Apple II Video Overlay Card (VOC), allowing you to show the video on the computer monitor—you can have buttons, text items, and other objects on the same screen as the video. If you do not have the VOC, you can still control a laserdisc player, but the video will have to be displayed on a separate monitor.

The player is attached to the Apple IIGS modem port via a "CC-04" cable, available from Roger Wagner Publishing. Make sure your "Slots" setting in the Control Panel has Slot 2 set to "Modem," and the "Modem" setting in the Control Panel has the "Baud" value at 4800. (After changing the Control Panel settings, be sure to reboot the computer to insure that the new values are used.)

If you do have an Apple II Video Overlay Card and wish to mix laserdisc video and computer graphics on the same screen, run the video out signal from the laserdisc player to the "Video In" jack on the VOC. (Be sure your Apple IIGS monitor is plugged into the RGB port on the back of the Video Overlay Card, and not in the RGB port on the back of the computer.) You may also want to run the "Video Out" from the VOC to a large screen television or even a VCR, to record the computer's display.

The Video Overlay Card does not control the audio of a laserdisc player, so you need to be certain to attach external speakers to the laserdisc's audio out jacks, on the back of the player.

HyperStudio does not automatically control VCRs, or other video sources, but it can (in conjunction with a VOC) use their video output.

Unidisk Drives

Important: the following information applies only to older, *white* 3.5" drives. The newer Apple 3.5" inch drives do not have this problem. Newer drives are gray (also called platinum).

Occasionally, the older drives will not know when a disk has been removed. Specifically, imagine the following scenario: you are working in HyperStudio. For some reason, it prompts you to insert a 3.5" disk. You already have another disk in the drive. You remove it, insert the correct disk, and press Return (or click "OK"). HyperStudio appears to not recognize the new disk, and asks you to insert it again.

The problem is that the computer still thinks the original disk is in the drive. The only solution is to remove the original disk and press Return (or click "OK") *with the disk drive empty*. This forces GS/OS to see that there's nothing there. Then, insert the correct disk and press Return (or click "OK") again.

We have been assured by Apple that this problem has been solved in the latest revision of the system software, but we've seen it still happen occasionally, so we're including the workaround here.

chapter

3

Reference

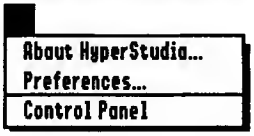
This section describes each of HyperStudio's menu items in detail. If you're looking for a description of exactly what a command does, this is the right place to check.

Reference

This chapter describes each of the menus of HyperStudio. Each command on the menus is explained. The menus (and commands) are covered from left to right, top to bottom, as they appear in the menu bar.

❖ Shortcuts and Tips

Menu menu items have keyboard equivalents. This means that you can press a combination of keys to bring up the menu command. When you see a keyboard equivalent here in the reference or in a menu, it will appear as "⌘-N." This means that you should hold down the Apple key, tap the "N" key, and then release the Apple key. (Note that on your keyboard, the Apple key may be called the command key, or may be labeled with the "propeller" symbol.)



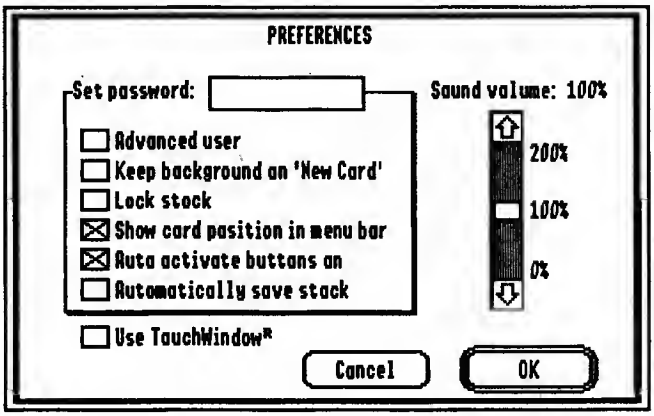
Apple menu

The Apple menu contains options to tell you about HyperStudio, and for you to control certain aspects of the program. It also contains all of your new desk accessories.

About HyperStudio...: This command opens a dialog box, telling you the authors of the HyperStudio program and the version number of the software.

Preferences:

Preferences lets you control many aspects of HyperStudio's operation. Many of the preference settings are saved in each stack, and may vary between stacks. The stack must be saved back to the disk after changing preferences if you want the changes to be in effect the next time the stack is used.



Set password: By entering a password into this field, all of the preference options below it (in the gray-outlined area) will be disabled and can't be changed. This is primarily useful for disabling access to the "Lock Stack" setting (see below). If the password field is blank, then all the options are available when Preferences is chosen. However, if you enter a password and exit the Preferences dialog, the password will be required the next time the Preferences dialog is opened. When re-entering the password, pressing Return is not required: when the last letter of the correct password is typed, the dialog box options will instantly become active.

To get rid of a password in a stack once you've added it, type in the password correctly, verify that all of the options are available (and that they are no longer dimmed), then erase the password edit line (press the Clear key) before clicking "OK".

A short sermon on setting a password on your stack: in general, there is no good reason to set a password on your stack. It limits the amount of flexibility that the user has and causes them to become frustrated if they simply want to browse around in your stack without using buttons (for example). While setting a password is a useful feature when you really need to lock the user completely out of your stack (such as in a testing stack, where the correct answers are stored elsewhere in the stack), we don't recommend it for everyday stack use and request that you please refrain from setting a password on your stack.

Sound Volume: This scroll bar lets you control the overall volume level of the stack in memory.

A sound in a stack stack created using the computer's speaker will be quite loud if you are using an amplified external speaker. Likewise, a stack that was set up using an external speaker may be too quiet using the computer's built-in speaker.

The volume in HyperStudio is controlled by a combination of factors. The Control Panel volume is figured into the overall sound volume in all HyperStudio stacks (this is the *system* volume). If you want to change the playback volume in a particular stack, use this scroll bar. This sets the *stack* volume. If you want the modified volume of the stack to be permanent, remember to save the stack back to disk. The volume level of a particular sound is selected in the HyperStudio Tape deck when a sound is first added to the stack.

☒ **Advanced User:** By marking this, you tell HyperStudio that you would like more control over how the data for objects on each card will be stored and accessed. Normally, HyperStudio embeds the data for each object in the stack, and doesn't ask how you want the data accessed. This is less confusing to a new user, and it creates fast and easily transportable stacks. However, if you are an Advanced User, you have options about how you want the object data accessed by HyperStudio.

When "Advanced User" is marked, a number of new features of HyperStudio are available. These are described below underneath the "Advanced User" heading.

The text of the second checkbox changes if "Advanced User" is checked. If "Advanced User" is not checked, it reads "Keep background on 'New Card'." Otherwise, the option is "Add new cards to group." Both are described below.

☒ **Keep background on 'New Card':** With this marked, HyperStudio keeps the same background that the current card has for any new cards added to the stack (with "New Card," in the Edit menu). If you are using one background (such as a notebook page) on many different cards in a stack, then you want to make sure this option is checked. That way, when you choose "New Card," the background will automatically be used on the new card. Since there may be no visible sign a new card has been created (particularly if you have no objects on the old card), you will probably want to have "Show card number in menu bar" turned on (described below).

When this option is not checked, new cards are erased to the current background color.

✔ **Advanced User**

If "Advanced User" is checked, the second preferences option will be "Add new cards to group."

Objects (graphics, buttons, and text items) can be designated as belonging to a group. They are automatically shared on all group cards. If the "Add new cards to group" checkbox is marked, any new cards that you create will be part of the *current* group of cards.

This means that, if you don't have this option checked, and you create five new cards, they will all be part of a group. As a group, they all share the same background, and any objects on the cards that are group objects will be shared among all of the cards in the group.

Changing the background on any card in the group will change the background on all of the cards. Similarly, changing a group object on any of the cards will affect it on all of the cards in the group. (Note that text items are group objects but the text in them is not; the attributes of the item, such as the position and color, will be repeated on all group cards, but the contents will not. The one exception to this rule is "title fields," discussed more in-depth in the "Add a Text Item..." section, below.)

If you choose to remove a card from a group (by turning off the "Group card" checkbox, in the "Card Info..." dialog box), it will be removed from the group, but will remain the same on the screen. All of the objects on the card that were group items will still be on the card, but they will no longer have their "Group Item" attributed checked. (If you uncheck the "Group Item" box on an object, it is no longer shared with all of the cards in the group, and becomes a non-group object on that card.)

New items that you create on a card, by default, do *not* belong to the current group, and you must specifically mark the "Group item" check box in the item's "Features..." box before it will be part of the current group.

Group cards are very useful when you wish to have common elements (such as a background or button) shared across many cards without using up memory for each. For example, a phone book stack could have a forward, backward, and home button on every card as a group item. Similarly, all of the cards in the group could share the same background and text items, for entering a name and phone number.

☒ **Lock Stack:** This option allows you restrict access to your stack. When you lock your stack (by marking this checkbox), all of the menus will be removed from the menu except for File, Edit, and Move. The File menu will only allow the user to start a new stack, open a stack, save a stack, save a screen, print a screen, or quit. The Edit menu will only have the standard editing options ("Undo," "Cut," "Copy," "Paste," and "Clear"), plus "New Card." Finally, the Move menu will contain only the standard stack navigation commands ("Back," "Home," "First," "Previous," "Next," and "Last"), plus "Find Text..." Additionally, users will not be able to enter the SimpleScript debugger by holding down the Apple key when a program executes. Unlocking the stack restores the complete menu bar. Since this option is available via the Preferences dialog box, you may wish to set a password in Preferences so that users can't get to the "Lock stack" checkbox without knowing the password.

☒ **Show card number in menu bar:** Use this to display the *position* of the current card within the stack at the right end of the menu bar. Note that the number displayed is *not* the card's ID number.

This is useful when creating and editing stacks. You may want to turn this off before creating the "final" version of your stack, so that the end user is not distracted by card numbers.

If the card number is shown in bold type, it indicates that the card you are on is part of a group.

☒ **Auto Activate Buttons On:** Auto-activate buttons are buttons that activate themselves after a set amount of time. However, you don't want them going off while you're working on a stack. By turning this option off, no auto-activate buttons will be triggered unless you manually click them. When you have finished creating or editing your stack, mark this checkbox again (so that auto-activate buttons will work as you have designed them), then save your stack. Note that auto-activate buttons are automatically turned off when you select a paint or editing tool, and are not turned back on until you select the Browse tool and move to another card.

☒ **Automatically Save Stack:** When this checkbox is on, HyperStudio will automatically save your stack any time that you are about to lose changes that you have made to it. (For example, right before you leave the program, or load another stack.) You will still need to name your stack the first time that it is saved.

Marking this option is the same as *always* answering "Save" to the standard "This stack has changed, do you wish to save it?" dialog box. You may wish to use this option sparingly; otherwise, any experimental changes you may make to a stack will be saved automatically when you don't want them to!

☒ **Use TouchWindow®:** This option allows you to use HyperStudio with the TouchWindow, by Edmark Corporation. The TouchWindow is a very useful add-on for HyperStudio: it allows you to touch the screen and navigate HyperStudio stacks the same as you would with a mouse.

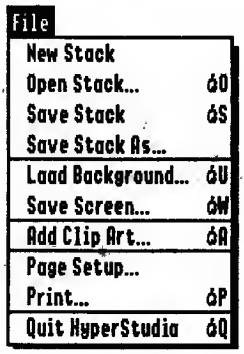
When you mark this checkbox and then click "OK" in the Preferences dialog box, it assumes you have a TouchWindow attached to your computer. At that point, the TouchWindow needs to be calibrated, and HyperStudio will prompt you to touch the upper-left and lower-right corners of the active screen area. (This does not include the monitor border area.) Avoid dragging your finger as you touch each corner, but don't poke at the corner: firmly press on the corner and hold it there until the computer beeps (indicating that you may continue). Remember, you are trying to tell the TouchWindow the location of a specific point (the corner of the screen). If the TouchWindow calibration is not correct the first time you do it, you may recalibrate it by unchecking "Use TouchWindow" in the Preferences menu, exiting and re-entering Preferences, and then marking "Use TouchWindow" again.

HyperStudio's TouchWindow capabilities are *not* stack-specific—calibration and activation does not have to be repeated for different stacks. Once activated, the TouchWindow will stay active until you exit HyperStudio. Additionally, HyperStudio will remember your TouchWindow calibration settings in any stack you save while the TouchWindow is on. This way, you can calibrate the TouchWindow once, and then

designate a stack as a "TouchWindow stack" by simply loading it and saving it with the TouchWindow active.

A stack that has been designated as a "TouchWindow stack" will still function normally if a TouchWindow is not plugged into the machine. It will automatically turn on a TouchWindow connected to the computer when it is loaded, and when a different stack is loaded, will turn the TouchWindow off (unless you've manually turned it on via the "Use TouchWindow" checkbox in the Preferences dialog).

For more information on the TouchWindow, please write the Edmark Corporation at P.O. Box 3218, Redmond, WA 98073-3218, or contact them by phone at (800) 426-0856.



File menu

This menu holds options that allow you to work with disk files and your printer. You'll find ways to save and load stacks and screens.

New Stack: This command starts a new stack, containing a single card with a white background. If you are in a stack that has been changed since the last time it was saved, HyperStudio will ask if you wish to save your changes.

Open Stack... (ó-O): This opens an existing stack. It can be a stack you made or one created by someone else.

When you load (or save) a file, you must identify it by name and location. This is done with the standard file dialog, which appears whenever you open a file, or save a stack for the first time. It is discussed more in chapter 2, "Setting Up."

The standard file dialog starts in the directory where HyperStudio is located. The files displayed will be of the type you are trying to load—that is, if you have told HyperStudio that you want to load a stack, only stacks (and folders) will be displayed. Similarly, if you want to load a graphic, only graphic files are displayed.

✔ Advanced User

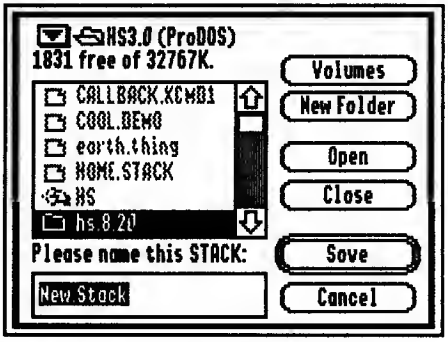
If you're an advanced user, when you tell HyperStudio to load an icon, sound, New Button Action, or transition, all files that may contain an icon, sound, NBA, or transition are displayed as well. This is so you can take the icon, sound, NBA, or transition from another program rather than a dedicated "library" file.

There are four buttons to the right of the file listing. The "Volumes" button displays all of the volumes online and allows you to select one. The "Open" button allows you to open a folder or file by highlighting it and then clicking "Open." (A shortcut for opening a folder or file is to double-click the folder or filename.)

The "Close" button lets you back out of the current folder. Finally, the "Cancel" button gives you a way to completely cancel the operation.

Save Stack (ó-S): With the Save Stack command, you instruct HyperStudio to write a copy of your stack out to disk. If you haven't named your stack yet, HyperStudio will automatically go to the "Save Stack As..." dialog box.

Save Stack As...: When you choose "Save Stack As...," the standard file dialog is shown.



Some extra functions are present that aren't available when you select "Open Stack..." The most important new item is the box in the lower left corner of the window—this is where you name your stack. (If your stack hasn't been named yet, the default name shown will be "Untitled".) Above this box, all of the files in the current directory are shown, in dimmed text. This is so you can see which names are already used.

The second new item is the button marked "New Folder." This allows you to create a new folder for your stacks by typing the name for the new folder and then clicking the "New Folder" button. The new folder will automatically be created, and you will be shown the contents of the new folder. (As a result of this, the list of files will be empty, since there's nothing in the folder yet.)

To save your stack after giving it a name, simply press Return or click the "Save" button.

Once a stack's name and location are saved, you can save your work with just the "Save Stack" command—HyperStudio remembers the stack's name and location, so you don't need to specify them again.

A good habit to develop is pressing ⌘-S (the keyboard shortcut for "Save Stack") any time that you pause in your stack development, or any time you make a large change to your stack—for example, after drawing a complex background or editing a large text item.

Use **Save Stack As...:**

- When saving a stack for the first time. HyperStudio needs to know a name for the stack and a location on the disk where you want it stored.
- Saving an existing stack (whether or not you've made changes) under a new name or to a new location.

Use **Save Stack:**

- When saving changes to a stack without changing the name or location on the disk.
- Before making changes, when you are unsure of the results. You can always reload the last-saved version.

Load Background... (⌘-U): Think of each card as a background graphic with objects placed on it. Even a blank card has a background, which is white. Graphics, text, or buttons can be placed on each background; it's simply a static framework on which the objects are presented.

Backgrounds can contain images that add to the visual appeal of the cards. A background could look like the pages of a book, for example, or a background could be a solid color. A stack can have as many different backgrounds as it has cards. A stack can also have one background, shared among all of the cards in the stack.

The "Load Background" command loads a 256-color or 320- or 640-mode, Super Hi-Res picture. (Note that 640-mode is the preferred format, however.) HyperStudio uses

that graphic as the background for the current card. You can change the background by simply using the paint tools. You can use 8/16 Paint, Paintworks Gold, Platinum Paint, DreamVision, or almost any 256-color, or 320- or 640-mode, Super Hi-Res drawing program to create backgrounds. For the vast majority of your work, however, you'll probably find it easiest to use HyperStudio's built-in paint tools.

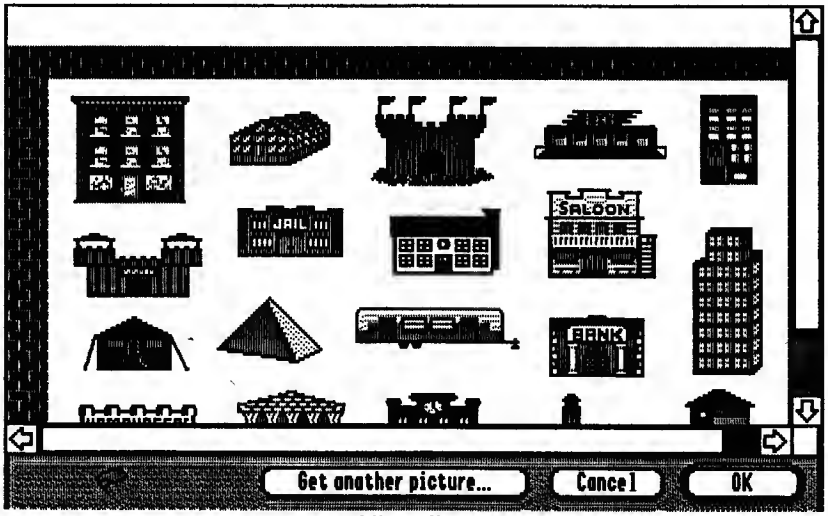
When you make a new card, HyperStudio will use the background on the current card with the new card if the "Keep background on 'New Card'" (novice user) or "Add new cards to group" (advanced user) checkbox is not marked. (For more information on this feature, see the discussion on group items in the section on "Preferences...," above.)

Save Screen... (Ó-W): The Save Screen command saves whatever is on the screen as a Super Hi-Res picture. Once saved, it can be used as a background or graphic in other cards or stacks by simply using the "Load Background" command, above. (Note that these files are saved in the standard Apple Preferred Format, so your favorite paint program should have no problem loading them.)

Screens can be saved just as you see them, with graphics, buttons, text items, and other objects visible as they appear. To save the screen without those objects, you can use the "Hide Items" command, under the Options menu.

Add Clip Art... (Ó-A): This command allows you to select part of a graphic from disk and paste it onto the background of the current card.

When you choose "Add Clip Art...," you will see the standard file dialog. It will show the file names of graphic images that you have. After selecting a file, a new window will open showing the graphic page.



adding clip art

Use the scroll bars to see all the art on the screen. To see other clip art files, click the "Get another picture..." button. You may open any 256-color or 320- or 640-mode graphic file, or a Finder icon file.

Notice the buttons on the bottom of the "Add Clip Art..." window. The icon on the left denotes the selection tool not in use. Click it to change between the selector tool and the lasso tool. (Clicking this icon causes the tool displayed to be used: if the selector tool is shown, then you are currently using the lasso tool, and vice-versa.) For more information on using the selector tool and lasso tool, see their descriptions in the Tools menu, later in this chapter.

The selector tool can select a rectangular chunk of graphics from the page, while the lasso tool lets you draw a line around the graphic you want. With the lasso tool, HyperStudio copies just the graphic selected, leaving the background behind. The selector tool copies the entire selection that you enclose, regardless of the contents.

If you make a mistake while selecting a graphic, simply try again. When you have selected an image, click "OK" to continue.

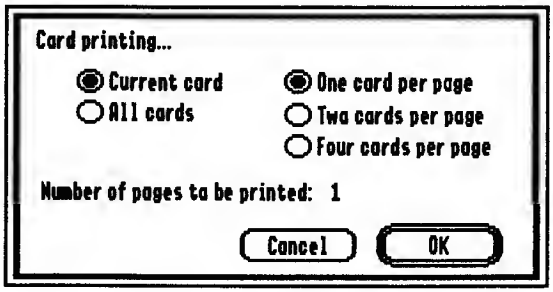
At this point, the clip art window will disappear, and the current card will re-appear, with your selected image floating in the middle of the current card. Put the mouse pointer over the middle of the image and drag the image to wherever you want it. Note that, because the clip-art you are adding will become a part of the background, all objects (such as buttons, text items, and graphics) will be drawn *over* the image that you are positioning.

Once you have the image where you want it, simply drop it in place by clicking away from it. (Note that, if you've used the selector tool and the image is still selected, you can use "Flip Horizontal" or "Flip Vertical" from the Edit menu at this point.) Once you click away from the image, it will be "pasted" onto the background and will no longer be highlighted.

Page Setup: With this option, you can set which type of paper your printer is using, which way things are printed on the page, print sizing, and whether to reduce the printed image. Usually the settings are for standard U.S. Letter size paper, normal vertical sizing and a vertical ("portrait") printout. The printer effects checkboxes are normally not marked, with no 50% reduction and no gaps between pages.

Print... (⌘-P): Use this to print the current card or the whole stack.

You can print one card per page, two cards per page, or with 50% reduction, four cards per page. HyperStudio computes how many pages of paper will be printed.



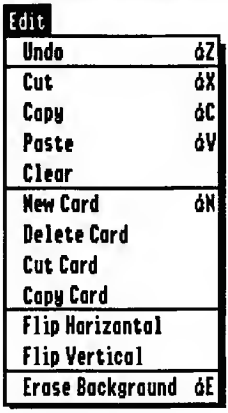
After selecting which cards you want printed, the standard print dialog box appears. Choose how many copies you want. Ignore the options for which cards you want to print, because you already set this.

The printing can be cancelled at any time by pressing Apple-period.

Note: if you run out of memory while printing, try using the run-time version of HyperStudio (`HS.Sys16`)—it occupies less memory than the regular version of HyperStudio, but has the same printing functions.

Quit HyperStudio (⌘-Q): Choose this to close the current stack. It will return you to the program that launched HyperStudio. Note that, if you started HyperStudio by booting the HyperStudio system disk, you will be returned to HyperStudio again (since there was no launching program). In this case, it is okay to press Control-Command-Reset to reboot your computer, after inserting another program's boot disk.

If the current stack has changed and has not been saved, HyperStudio will prompt you to save your changes before leaving.



Edit menu

This menu lets you cut, copy, and paste objects, parts of the background, or entire cards.

Undo (⌘-Z): Use this when you change your mind about certain editing actions. For example, if you cut a button from a card, and then change your mind, you can choose "Undo." The button will be put back as it was before the cut. If you paste a graphic onto the wrong card, choosing "Undo" takes the graphic off. Note that "Undo" only reverses your most recent action, and that not all actions can be undone.

Cut (⌘-X): To cut an object (graphic, text or button) or selected text, just select it, then choose "Cut" from the Edit menu. This removes the object from the card, and puts it on the clipboard. Once an object is on the clipboard, you may use the "Paste" command to make a copy of it.

An object stays on the clipboard until you quit HyperStudio, or place something else on the clipboard with a "Cut" or "Copy" command.

You can also cut portions of the background for pasting somewhere else—just use the lasso or selector tool to select the part of the background and pick "Cut" from the Edit menu.

Copy (⌘-C): "Copy" places a copy the selected object or portion of the background on the clipboard. It is the same as "Cut," except that the original object (or selected area of the background) is not removed.

Paste or **Paste Card (⌘-V):** Choosing "Paste" moves a copy of the object from the clipboard to the current card.

If an object is on the clipboard, after selecting "Paste" from the menu, it will be put on the current card and selected. Use the mouse or the arrow keys to place it where you want, then click the mouse anywhere outside of the object to drop it into position. Because the item stays on the clipboard until something else is copied or cut, you may paste copies as many times as you like.

If a card is on the clipboard, then this menu option will read "Paste Card." Selecting it will cause a copy of the card to be inserted after the current card.

✔ Advanced User

Text items, buttons, and graphics can be designated as group items. All items on a card you cut or copy will be copied to the clipboard, and will still be on the card when you paste it. This way, you do not have to make the same things on each card. A card pasted anywhere in the same stack will maintain changes made to grouped items. Pasting a group card in another stack will keep the group items with it, but will not maintain the link to the original stack. (This way, the card will still have the same objects that it did in the original stack, and the objects will still be designated as group objects, but they will not be part of any group since the card will be independent.)

Clear: This command is similar to "Cut," except that it completely removes the selected item (or selected text or portion of the background) and does not change the clipboard in any way.

New Card (⌘-N): This command will create a new card in your stack.

If **Keep background on "New Card"** is not marked in the Preferences dialog, the new card will start with a blank background.

If **Keep background on "New Card"** is marked in the Preferences dialog, the background of the current card is copied onto the new card. Objects on the current card are not copied.

✔ **Advanced User**

New Card will create a new card in your stack, and can optionally copy the background and objects from the current card to the new card (if you add the new card to a group).

If **Add new cards to group** is not marked in the Preferences dialog, the new card is erased to the current background color.

If **Add new cards to group** is marked in the Preferences dialog, the background and the objects on the current card are shared with the new card. They are considered group objects; be sure to see the discussion of group objects in the "Preferences..." section, below.

Delete Card: The "Delete Card" command completely removes a card from the stack. Note that if you attempt to delete a card which is pointed to by buttons elsewhere in the stack, you will be asked if you wish to delete those buttons, as well. If you leave a button like that in place, the user will receive an "Unable to find card" message when it is clicked.

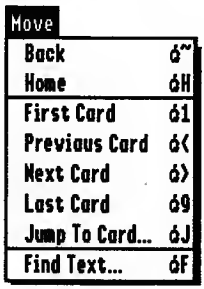
Cut Card: This command moves an entire card to the clipboard, removing it from the stack. The card will remain on the clipboard until you cut or copy something else.

Like "Delete Card," if you attempt to cut a card which is pointed to by buttons elsewhere in the stack, you will be asked if you wish to delete those buttons. If you leave the buttons, the user will receive an "Unable to find card" message when the button is clicked.

Copy Card: This is similar to the "Cut Card" option, except that it only places a copy of the current card on the clipboard and does not remove the card from the stack.

Flip Horizontal, Flip Vertical: After selecting an image with the selector tool, choosing this option will cause the image to be flipped in the indicated direction. (You can also use this option before dropping clip-art in place.) This option is not available for lasso-selected graphics.

Erase Background (⌘-E): This erases the current background to one of the available colors or patterns. (See "Set Background Color..." in the Options menu.)



Move menu

The options on this menu provide you with a method of moving between the cards in your stack, whether or not you already have any buttons in place.

Back (ó-~): This takes you back, one card at a time, through the cards viewed, up to 16, since starting HyperStudio. (These 16 cards comprise the "back list.") It goes back through the cards in the order you viewed them, even if they were in different stacks!

Home (ó-H): The home stack is an important location in the HyperStudio environment—it's a "home base," where you usually start from and frequently go through when moving from stack to stack. By selecting "Home," you are quickly taken to the last stack loaded that was named `Home.Stack`.

Note that many stacks have home stack icons. Clicking on a home stack icon will return you to the home stack, and is the same as selecting "Home."

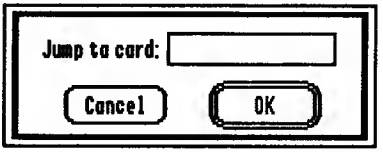
First Card (ó-1): This takes you to the first card in the current stack.

Previous Card (ó-<): This takes you to the previous card in the current stack. If you are already on the first card in the stack, this takes you to the *last* card in the stack. (This is called the "wrap-around" effect.)

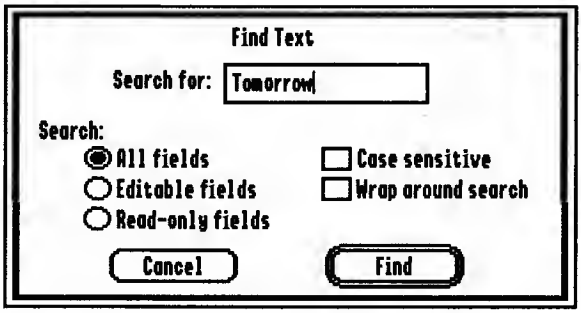
Next Card (ó->): This takes you to the next card in the stack. If you are already on the last card in the stack, this takes you to the first card in the stack.

Last Card (ó-9): This takes you to the last card in the stack.

Jump to Card... (ó-J): Selecting this opens a dialog box which allows you to quickly move to a card by specifying the card's name (if you enter a name) or position (if you enter a number). Choose "Card Info" from the Objects menu to see if a card has a name, or to give it a name.



"Jump to Card" is disabled if a stack is locked, has a password set, and the menu bar is not being displayed. (That is, you can't press Apple-J to access "Jump to Card," in that case.)

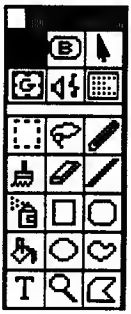


Find Text... (⌘-F): With this command, you can search for a word or phrase in all of the text items in the current stack. HyperStudio starts searching from the first text item on the current card.

You can search the editable or read-only items on each card, or both. Clicking the "Case sensitive" checkbox will limit the search to text that matches the letters exactly you typed, while clicking the "Wrap around search" checkbox tells HyperStudio to search the entire stack, not just from the current card to the end of the stack. Click "Find" (or press Return) to start the search. To search for the same word or phrase again, press **Shift-⌘-F**.

Note: The "Find Text..." command only works on text items. It cannot "see" words painted by the text painting tool from the Tools menu.

When using "Find Text..." in a locked stack with a password set (in the Preferences dialog box), text can not be found past of the current card.



Tools menu

The Tools menu has two groups of tools. The upper group represents editing tools: they are for screen items like text, graphics, and button objects. The lower group is composed of paint tools, used to create or edit the current background.

If you want to keep the Tools menu handy on the screen, you can "tear it off." Simply drag the mouse through the bottom or the sides, and then release the mouse button wherever you want to leave the menu. Drag the "torn off" Tools menu by the top gray bar to move it around. Click in the upper-left corner to make it disappear.

Browse Tool: Clicking on this tool gives a hand icon. It is used to browse through a stack and trigger buttons. This is default tool you see when HyperStudio is started, or when a stack is loaded.

Note that as you get more objects on the screen, and they all are re-drawn as you click the mouse, you find the object doesn't appear to always respond to a double-click. If this happens, try double-clicking a little more slowly. (A slower double-click is easier for the system to detect.) You can also change the double-click speed that the computer is looking for in the Control Panel.

❖ Shortcuts and Tips

Shift-Tab will toggle between the Browse and Edit tools.

Pressing the Command and Option keys at the same time will display all invisible buttons.

Button Tool: This is used to highlight buttons for editing, moving or resizing. (Invisible buttons are shown surrounded by rectangles.) To move a button, first click it to select it, then drag it to a new location (or use the arrow keys to move the button in very small increments). For more information, see "Button Info...," in the Objects menu, described below.

❖ Shortcuts and Tips

Double-click a button to change its appearance.

Double-click a button while holding down the Apple key to change its actions.

Select a color from the Colors menu to change a button's color.

Select a color from the Colors menu while holding down the Option key to change a button's background color.

Use "Set Text Color" in the Options menu to change the button's text color.

Edit Tool: This tool is a general purpose editing tool and is used to select items for editing.

To reposition an object, click the item to select it, then drag it to a new location (or use the arrow keys to move the object in very small increments).

To resize an object, click the item to select it, then drag the corners of the enclosing rectangle to change the size. If you wish to edit an object, simply select it by clicking it and then pick "Item Info" from the Objects menu. (Note: the first option in the Objects menu will not really be "Item Info," but will instead reflect the type of object selected: it will read with "Button Info...," "Graphic Info..." or "Text Info...")

There are three "specialized" editing tools described below: the Graphic Tool, the Sound Tool, and the Text Tool. These are included so that you may edit graphic objects, buttons with attached sounds, and text objects (respectively) without other objects getting in the way. Selecting one of these specialized tools will cause any objects that the tool is *not* capable of editing disappear from the screen. (To make all of the objects reappear, simply select the Edit or Browse Tool.)

❖ Shortcuts and Tips

Double-click an object to go directly to its "Item Info..." dialog.

Shift-Tab will toggle between the Browse and Edit tools.

Graphic Tool: One of the specialized editing tools, the Graphic tool will let you edit all the graphic objects on the current card. Use it by clicking once on a graphic item to select it, and then you can resize it, move it, or cut, copy, or delete it. For more information, see "Graphic Info...," in the Objects menu, described below.

❖ Shortcuts and Tips

Double-click a graphic to go directly to the "Graphic Info..." dialog.

Hold down the Option key to change the graphic to its original size, after resizing it.

Sound Tool: This tool will let you edit all the buttons on the current card that have sounds attached. Use it by clicking once on a button to select it, and then you can resize it, move it, or cut, copy, or delete it. For more information, see "Button Tool," above.

❖ Shortcuts and Tips

Double-click a button to go directly to the "Button Info..." dialog.

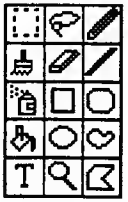
Text Tool: Selecting this tool displays all text items on the current card. Click a text item to resize it or drag it, or to cut, copy, or delete it.

To edit the *text* in a text item, use the Browse tool and click the text in the item. Note: this assumes that your text is not read-only. If it is, you need to change its access by double-clicking the text item with the text tool. Disk-based (extended) text items are always read-only.

To change the *appearance* of a text item (or its attributes, such a read-only), use the Text tool and click the item. Choose "Text Info..." in the Objects menu to make changes. You may also change whether it is a read-only or changeable text block. Disk-based (extended) text items are always read-only. Having selected an entire text item, you can also change the text color, background color, and text style by using the appropriate items in the Options menu. For more information, see "Text Info...," in the Objects menu, described below.

❖ Shortcuts and Tips

Double-click a text item with the Text Tool to go directly to the "Text Info..." dialog.



Paint Tools

The remainder of the tools in the Tools menu are for editing a card's background image. You can load any 256-color, or 320- or 640-mode, Super Hi-Res screen as a card background or a piece of a Super Hi-Res screen as a graphic object.

If you change your mind or make a mistake after painting on the card background, just choose "Undo" in the Edit menu (or type ⌘-Z). You can only undo your most recent painting action—for example, if you draw three lines with the Paintbrush tool, then select "Undo," only two lines will remain.

Some paint functions may be modified with a key combination—holding down the Shift key when painting with the oval tool, for example, will cause the tool to produce only circles. Additionally, if you have "torn off" your menu to keep it handy on the screen, double clicking many of the icons will call up options. See appendix A, "Tools Reference," for a complete list of all of the tools and their associated "modifiers."

Selector Tool: This tool allows you to select a rectangular area to be cut, copied, deleted, or moved. When this tool is active, the cursor changes to a cross-hair. To select an area, simply position the cursor at one corner of the desired area, press the mouse button, and drag the mouse while holding down the mouse button. After you release the mouse button, a blinking dotted line will indicate the selected area.

At that point, you can choose a number of options from the Edit menu, including "Flip Horizontal," "Flip Vertical," "Cut," "Copy," and "Clear." Cutting or copying the selected area will place it on the clipboard where you can paste it on the current card, another card, or even another stack. You may paste the image as many times as you like; it remains on the clipboard until you cut or copy something else. If you cut the graphic out of the background, the selected area is replaced by whatever color is under the upper-left corner of the selected rectangle.

If you want to move the selected area, put the mouse inside the rectangle. The cursor will change to an arrow. Hold down the mouse button and drag the mouse. When you release the mouse button, the image will be placed at that location. The area under the moved rectangle is replaced by the background color.

If you select an area and then change your mind, click the mouse outside the selected area to remove the selection box.

❖ Shortcuts and Tips

Double-click the Selector tool icon to select the entire background.

Hold down the Apple key while dragging the mouse to stretch or shrink the selected area.

Hold down the Option key while dragging a selected area to create a copy without moving the original.

Press the Delete key to delete a select area.

Lasso: This is similar to the Selector tool. You use it to outline an image on a solid-colored background. After selecting the lasso tool, you draw a free-hand line around the object to be selected. When you release the mouse button, the color under the cursor where you started the lasso is used as the background color, and only the parts of the image different from that background are selected with an outline.

Once selected, the item can be cut, copied, resized, or moved, just as you would with the Selector tool.

❖ Shortcuts and Tips

Double-click the Lasso tool icon to lasso the entire background.

Hold down the Apple key while dragging the mouse to stretch or shrink the selected area.

Hold down the Option key while dragging a selected area to create a copy without moving the original.

Press the Delete key to delete a select area.

Pencil: With the Pencil tool, you can draw freehand in the currently selected color. (Select the color from the Colors menu.)

❖ Shortcuts and Tips

Double-click the Pencil tool icon to enter fat bits mode. (For more information on fat bits mode, see the information on the Magnifying Glass tool, below.)

Use the Colors menu to select a new color or pattern.

Hold down the Shift key to draw only straight lines.

Paintbrush: The Paintbrush tool uses the current brush shape to allow you to draw freehand in the current color or pattern. Select "Brush Shape..." from the Options menu to pick a new brush shape.

❖ Shortcuts and Tips

Double-click the Paintbrush tool icon to go directly to the "Brush Shape..." dialog box.

Use the Colors menu to select a new color or pattern.

Hold down the Shift key to paint only straight lines.

Eraser: The Eraser tool is a handy way to erase an area of the screen. (It's the same as drawing with a rectangular brush in the background color.) To use it, press and hold the mouse button while dragging over the area you want to erase.

You may set the color that the eraser uses by going to "Background Color...", in the Options menu. If you click the "Erase Now" button, the entire screen will immediately erase to the specified background color. (Selecting a color from the Colors menu while holding down the Apple key will also set the background color.)

❖ Shortcuts and Tips

Double-click the icon to erase the entire screen to the background color. (If you do this by accident, immediately select "Undo" from the Edit menu.)

Hold down the Shift key while dragging the mouse to erase in a straight line.

Line Tool:

This is used to draw straight lines in the current color or pattern. Press the mouse button where you want the line to begin. Drag the mouse. Notice that while the mouse button is down, you can move the end of the line anywhere. Release the button where you want the line to end.

❖ Shortcuts and Tips

Select "Line Size..." from the Options menu to pick a new line size. (You may also double-click the Line tool icon.)

Select "Draw Multiple" from the Options menu to leave lines on the screen as you drag the mouse.

Select "Draw Centered" from the Options menu to draw from a center point instead of the end.

Use the Colors menu to select a new color or pattern.

Hold down the Shift key to draw only straight lines.

Spraypaint Can: This tool paints a scattered pattern of dots in the current color or pattern. You can click repeatedly over an area while moving the mouse very slightly for a "repeat spray" effect, or drag it for a "scattered" effect. By moving it over an existing image you can create variations in shades—move the mouse quickly for a more diffused pattern, or slowly for a denser pattern.

❖ Shortcuts and Tips

Use the Colors menu to select a new color or pattern.

Hold down the Shift key to paint only straight lines.

Rectangle: This tool easily allows you to draw a rectangle.

Rounded Rectangle: This tool draws a rectangle with rounded corners.

To use either, press the mouse button and hold it down while you drag the rectangle to size. Release the button when the rectangle is the desired size.

❖ Shortcuts and Tips

Select "Draw Filled" from the Options menu to draw a solid rectangle. (When this item isn't marked, you only draw the outline of a rectangle. Note that you may double-click the icon to toggle the "Draw Filled" option on and off.)

Select "Draw Multiple" from the Options menu to leave lines on the screen as you drag the mouse.

Select "Draw Centered" from the Options menu to draw from a center point instead of a corner.

Select "Line Size" from the Options menu to change the size of the rectangle's outline.

Use the Colors menu to select a new color or pattern.

Hold down the Shift key to draw squares instead of rectangles.

Fill: This tool (shown as a tipped paint can with paint pouring out) is used to fill an *enclosed* area with a solid color or pattern. The tip of the cursor (the very end of the paint pouring out of the can) is where the fill originates.

Be sure to completely enclose the area you wish to fill; gaps will cause "leakage." If the fill covers a larger area than you intended, choose "Undo" from the Edit menu, then check the boundaries of your area. (Use the Magnifying Glass tool to help find small gaps.)

Oval: This is similar to the rectangle tools, except it draws an oval shape. The point where you press the mouse button represents the upper-left corner of an imaginary rectangle. It contains the oval being drawn. The point where you release the mouse button represents the lower-right corner.

❖ Shortcuts and Tips

Select "Draw Filled" from the Options menu to draw a solid oval. (When this item isn't marked, you only draw the outline of an oval. Note that you may double-click the Oval tool icon to toggle the "Draw Filled" option on and off.)

Select "Draw Multiple" from the Options menu to leave lines on the screen as you drag the mouse.

Select "Draw Centered" from the Options menu to draw from a center point instead of a corner.

Select "Line Size" from the Options menu to change the size of the oval's outline.

Use the Colors menu to select a new color or pattern.

Hold down the Shift key to draw circles instead of ovals.

Freehand Shape: This tool draws a closed freehand shape. You simply draw freehand, and the computer will create a closed shape. (A line will be drawn from the end point to the starting point if you do not close the shape.)

❖ Shortcuts and Tips

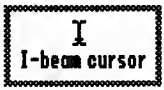
Select "Draw Filled" from the Options menu to draw a solid shape. (When this item isn't marked, you only draw the outline of the shape. Note that you may double-click the icon to toggle the "Draw Filled" option on and off.)

Select "Line Size" from the Options menu to change the size of the rectangle's outline.

Use the Colors menu to select a new color or pattern.

Text: This tool is used to paint text onto the background. Note that this is not the same as adding a text item: a text item is an independent object that can be moved and edited at any time, while the *painting* text tool, on the other hand, simply draws the text you type on the screen, where it becomes part of the background.

When you choose the text tool, the pointer changes to an I-beam cursor.



To type text, place the I-beam where you want to start typing. Click the mouse once. The I-beam will disappear and a blinking cursor will appear on the background.

The text will usually look best if it is drawn in black or white, but the quality also depends on the background color and the general design of the font you're using. In general, contrasting background and foreground colors, large type sizes, and legible fonts all contribute to a clear display.

You can change the type style at any time, even while typing. However, once you reposition the text cursor (by clicking the mouse in a new location on the screen) or

choose another paint tool, the text becomes part of the background image. Once there, you can change it only by painting over what you don't want.

❖ Shortcuts and Tips

To change the size, color, and font, select "Text Style" from the Options menu. (You can also double-click the Text tool's icon.)

Use the Colors menu to select a new color or pattern.

Magnifying Glass: This is a special mode for magnifying the background image. You can edit the screen dots on a pixel level, allowing you to make very minute changes to the display. (This is sometimes called a "fat bits" or "zoom" mode.)

When you select this item, the cursor will change to a magnifying glass. Click the center of the cursor over the area you want to examine. You will see a magnified image of the area of the screen in the rectangle, with a normal view of the same area in the upper-left corner of the screen. Your only painting tool in fat bits mode is the Pencil, which is automatically selected for you.

To return to normal view, select any other paint tool, or click the mouse in the small regular sized image.

❖ Shortcuts and Tips

Double-click the Pencil tool to quickly jump into fat bits.

Holding down the option key will change the cursor to a hand, which allows you to scroll the screen by simply dragging in any direction.

Clicking in the regular size image (in the upper-left corner of the screen) will take you out of fat bits mode.

Select a new color to paint with from the "Colors" menu, or hold down the Apple key and click a pixel to select its color.

Attempting to paint a pixel that is already in the current color will flip the Pencil tool over and use the eraser end: the Pencil will then "erase" in the background color. Releasing the mouse button immediately changes the Pencil tool back to normal.

Hold down the Shift key before dragging the mouse to constrain your drawing to only horizontal or vertical lines.

Polygon: This tool draws a closed polygon shape. Click at the start point. Click again wherever you want to add another "corner." When you release the mouse button, it will draw a line from the end point to the starting point, if necessary.

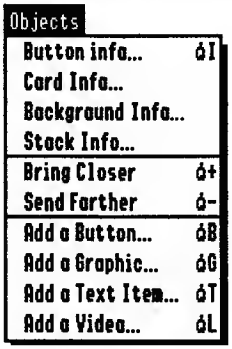
❖ Shortcuts and Tips

Select "Draw Filled" from the Options menu to draw a solid polygon. (When this item isn't marked, you only draw the outline of a polygon. Note that you may double-click the Polygon icon to toggle the "Draw Filled" option on and off.)

Select "Draw Multiple" from the Options menu to leave lines on the screen as you drag the mouse.

Select "Line Size" from the Options menu to change the size of the polygon's outline

Use the Colors menu to select a new color or pattern.



Objects menu

This menu is used to add new objects to a card, and edit objects already on a card. It also provides information about the memory used by the current background, card, stack, or a specific object.

To get information on an object, use a specific tool (such as the Button, Text, or Graphic tool), or the Edit tool (the pointer) to select an object on the current card. Then, choose the first option in the Objects menu, which will reflect the type of object selected: it will be either "Button Info...," "Graphic Info..." or "Text Info..." All three of these are described below.

Button Info... (⌘-I): This shows the button name, style, any associated icon, and the foreground and background colors. This is the same as the "Add a Button" screen displayed when you created the button. For more information, please see chapter 4, "Adding a Button."

Whatever is entered on the edit line is used for the button name. Even invisible buttons can have names—though not displayed, their names are written out to the disk as part of the `HS.Test.Results` file for visible or invisible buttons (if the stack is a test effect stack). Names also allow SimpleScript and NBAs to manipulate items.

Clicking the "Actions..." button brings up a dialog box defining the button's actions: where it connects to, any tasks it should perform (such as playing a sound or video), and other information. (This is the same as the "Button Actions" screen displayed when you created the button, and again, for more information, see chapter 4, "Adding a Button.")

Clicking "Icons..." shows the icons available for the button.

✔ Advanced User

Advanced users see an additional checkbox in the "Add a Button" dialog box—"Highlight." It controls whether or not the button highlights (reverses color) when it is clicked.

Additionally, there is another button in the dialog box, labeled "Features..." Clicking it opens the Item Features dialog box. This dialog box is divided into two halves.

The upper portion tells you the type of the button, the amount of memory it occupies (including all attached items such as sounds, animations, or script), the "owner" card (where the button definition is stored), the button's ID, and the button's position on the card.

The button's ID and position can be useful for certain SimpleScript commands, which require either a button's position or name.

The lower portion of the Item Features dialog box features three checkboxes:

☒ **Locked:** With this box checked, the button may not be moved or edited.

☒ **Group Item:** Any button with this item checked will show up on all cards within the same group. If you unmark this checkbox, the button will no longer be a group item and will appear only on the card you were at when you unchecked this box. Marking the box will cause the button to once again be shared with all of the cards in the group.

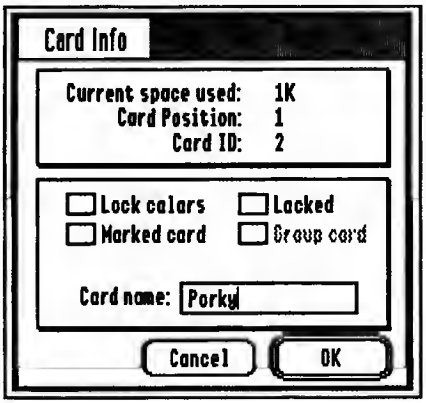
☒ **Hidden:** When the Hidden checkbox is marked, the entire button is invisible and may not be clicked. It will also not auto-activate. (It will show up, however, when you switch to the Edit tool or the Button tool.)

Text Info... (Ö-I): Selecting "Text Info..." will present you with a dialog box showing an example of your text item, its attributes, and the item's name. This is the same as the "Add a Text Item..." dialog box; for more information, see "Add a Text Item..." (in the Objects menu), described below.

Graphic Info... (Ö-I): This command displays a sample of the graphic object, along with presenting options to change the frame color, frame width, and the object's name. This is the same as the "Add a Graphic Object..." dialog box; for more information, see "Add a Graphic Object..." (in the Objects menu), described below.

Card Info...: When this command is selected, HyperStudio indicates the position of the current card in the stack, and how many buttons, sounds and graphics are on that card. It also reports how much memory is used by that card.

There are also several card options that you can toggle:



☒ **Lock colors:** Normally, when importing clip-art onto a card (with the "Add Clip-Art..." command, under the File menu), HyperStudio changes the card's colors to match the colors of the new art. With this box checked, the card's colors will never be changed.

☒ **Locked:** With this item checked, the card may not be cut or deleted.

☒ **Marked card:** When the "Marked card" checkbox is set and the user goes to that card, the card will be remembered as the most recent marked card. As a result, a button that returns to the last marked card will return here.

There's one final checkbox, "Group card," which is dimmed for non-group cards. It simply indicates whether or not this card is in a group. If the "Group card" checkbox is marked, it means that the background and objects are shared with other card(s) in the stack. In this case, you can remove the card from the group by simply unmarking this checkbox.

Background Info...: This shows how much memory the current background uses, and how many cards in the stack use that background. Backgrounds are always contained within the stack itself, and when the background is stored in the stack, it is compressed to use as little memory as possible.

Stack Info...: This command provides information about the entire stack. It includes the size of the stack in memory, and how many cards, buttons, text items and graphic objects are in the stack. It also reports the amount of memory that the stack currently occupies in the machine, how much space it requires on disk, and how much memory is available in the machine.

Since HyperStudio has the ability to temporarily remove sounds and graphic objects from memory if it doesn't need them, and retrieve them later, the amount of memory that the stack occupies in memory may vary from the amount of space it occupies on disk. (This could lead to a 200K stack occupying only 50K in memory.)

This also works in the opposite direction—a 2K stack that is nothing but disk-based text items could occupy 50K in memory, once all of the text items have been filled from disk and are stored in memory so that they may be displayed on the card.

Bring Closer (⌘-+): If you encounter two objects on the screen that overlap, this option will bring the selected object forward one layer. Note that group items are always behind non-group items. Pressing **Shift**-⌘-+ will bring the object all the way to the front.

Send Farther (⌘—): This command, similar to "Bring Closer" (above), will take a selected object and move it back one layer. Note that group items are always behind non-group items. Pressing **Shift**-⌘— (dash) will place the object all the way at the back.

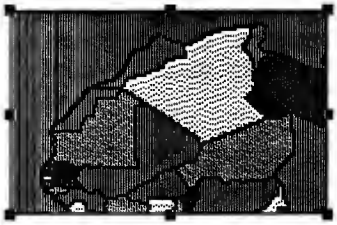
Add a Button... (⌘-B): With this command, you can add a button to the current card. See the next chapter, "Adding a Button," for complete details.

Add a Graphic... (⌘-G): This menu item is used to add a graphic object to a card. You will get the standard file dialog for choosing the graphic file you wish to use. If this is the first graphic you have added since starting HyperStudio, you'll get a second dialog box, with brief instructions. (If you are an Advanced User, you will not get this message.)

The graphic object that you add retains its own identity as an object on the card. This is in contrast to adding clip art to a card, where the clip art becomes part of the card's background. Graphic objects are best used when you have many cards that will use the

same graphic element. If you are just adding some graphics to a card, it is better to use the "Add Clip Art..." function in the File menu.

After you have chosen a disk file from which to get your graphic object, a rectangle will be displayed that shows a portion of the graphic screen you selected. This is sort of a "window" that you're looking through to see part of the graphic.



You have to decide how much of the image you want displayed on the card, and where you want it. Put the mouse anywhere inside the rectangle to change the cursor to a hand, then hold the mouse button down and drag the mouse to move the entire graphic image under the "window."

If you want to resize the graphic object's rectangle, use the mouse to grab any handle on the sides or corners and drag to adjust. The cursor will turn to a four-headed arrow when you are positioned correctly on the frame. If you hold down the Option key and attempt to drag a graphic object, it will return to the original size when it was placed in the stack.

If you want to move the rectangle to a new position, drag any part of the frame that is *not* a handle. The cursor will turn to a four-headed arrow when you are positioned correctly on the frame.

After you have positioned and cropped the graphic as you desire, click anywhere outside the object's rectangle (or press Return) to indicate that you are done. HyperStudio will display the Graphic Info dialog box,where you may enter a name for the graphic, and select the color and width of the graphic's frame.

To adjust the frame's width, click the up or down arrows. If you don't want a frame with your graphic, click the down arrow until the frame disappears.

Once a graphic object has been created, it can be scaled by using the Graphic Tool or the Edit Tool and simply dragging a corner of the object. To restore the graphic to its original size, hold down the Option key and drag one of its corners.

✔ Advanced User

Advanced users also have the option of selecting where the graphic object will be saved: click "Embedded" to save it as part of the stack, or click "Disk-Based" to leave it on the disk. If you choose to keep it on disk, HyperStudio will remember the pathname to the artwork that you used to get the graphic object, and how you cropped and positioned the graphic. (In other words, HyperStudio doesn't create a new disk file, but uses the same file that you did to make the graphic object.)

Additionally, there is another button in the dialog box, labeled "Features..." Clicking it bring up the Item Features dialog box. This dialog box is divided into two halves.

The upper portion tells you the type of the graphic object, the amount of memory it occupies, the "owner" card (where the object's definition is stored), the graphic object's ID, and the object's position on the card.

The object's ID and position can be useful for certain SimpleScript commands, which require either an object's ID or name.

The lower portion of the Item Features dialog box has four checkboxes:

☒ **Locked:** With this box checked, the graphic object may not be moved or edited.

☒ **Group Item:** Any object with this item checked will show up on all cards within the same group. If you unmark this checkbox, the object will no longer be a group item and will appear only on the card you were at when you unchecked this box. Marking the box again will cause the object to once again be shared with all of the cards in the group.

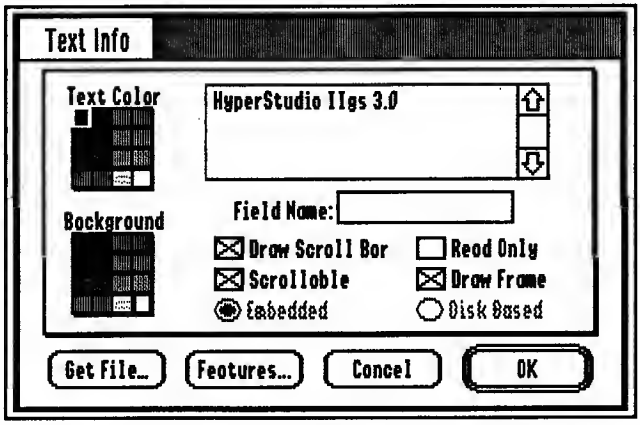
☒ **Hidden:** When the "Hidden" checkbox is marked, the entire graphic object is invisible. (It will show up, however, if you use the Edit tool or the Graphic tool.)

☒ **Draggable:** If this option is marked, the user will be allowed to drag the graphic object around the screen with the Browse tool. This can be useful for SimpleScript programs where you want the user to respond by dragging a graphic, rather than clicking a button.

Add a Text Item... (Ō-T): With this command, you are able to add text items to a HyperStudio stack.

After selecting this option, a rectangle will appear on the screen. You can drag from the center of the rectangle to position it on the screen, or drag from the corners to change the rectangle's size.

When you have the text object's rectangle positioned as you like it, click outside of the rectangle or press Return. You will be presented with a dialog box showing an example of your text item, its attributes, and the item's name.



The four attributes are:

☒ **Draw scroll bar:** Marking this checkbox will cause HyperStudio to draw a scroll bar on the right side of your text item, allowing the user to move vertically through the contents of the text item.

☒ **Read only:** By turning this checkbox on, you prevent the user from changing the text in your text item. (Note that disk-based text is always read-only, and will have the "Read only" checkbox dimmed.)

☒ **Allow scrolling:** With this option turned on, the user is allowed to scroll through your text item with the arrow keys or a scroll bar (assuming "Draw scroll bar" is marked, above).

Marking "Allow scrolling" and "Read only" and leaving "Draw scroll bar" unmarked is a perfect configuration for the Roll Credits NBA. (More information on the Roll Credits NBA can be found in appendix D, "Sample NBAs, Extras, and Transitions.")

☒ **Draw frame:** If this option is marked, HyperStudio will draw a thin frame around the boundary of your text item.

Which attributes you use depend on the the purpose of your text item. If you are creating single-line entries, such as in an address book stack, you will want each item to be non-scrolling, without scroll bar or a frame. If you're creating a text item that you want to masquerade as part of the background, you may wish to have it the object as read only, with no scrolling, frame, or scroll bar.

You can also choose the color of the text and background of the item. A sample window will show the color combinations.

Finally, you may decide if you want to use text from a file already on disk, or if you want to enter text directly. To use a file from disk, click the "Get File..." button. You'll see the standard file dialog, which allows you to choose an ASCII text file or AppleWorks Classic word processor file. The file will be loaded and the text will appear in the text object.

Whether text was created from a disk file or entered directly, you can still resize, reposition, or change the attributes of the text box. Just use the Text or Edit tool in the Tools menu, select the text item, and then choose "Text Info..." from the Objects menu.

✔ Advanced User

Below these four checkboxes are two dimmed radio buttons, indicating the location of the text in the text item (embedded or disk-based). The radio buttons do not become active until you after you select a file from disk (by clicking "Get File...").

Advanced users have another button in the dialog box, labeled "Features..." Clicking it brings up the Item Features dialog box. This dialog box is divided into two halves.

The upper portion tells you the type of the text item, the amount of memory it occupies, the "owner" card (where the text item definition is stored), the item's ID, and the object's position on the card.

The object's ID and position can be useful for certain SimpleScript commands, which require either a text item's ID or name.

The lower portion of the Item Features dialog box features four checkboxes:

☒ **Locked:** With this box checked, the text item may not be moved or edited.

☒ **Group Item:** Any object with this item checked will show up on all cards within the same group. In the case of a text item, it means that the text item's location and attributes (color and style) will be repeated on all group cards, but the contents of the text item will be independent. If you unmark this checkbox, the object will no longer be a group item and will appear only on the card you were at when you unchecked this box. Marking the box again will cause the object to once again be shared with all of the cards in the group.

There is one exception to the contents of a group text item not being shared across all cards: if you marking a text field "Read only" after creating it but *before* leaving the card that it resides on, it will become a "title field," and will be shared (contents and all) with all cards in the group. You can edit the object by making it non-read-only, changing it as you like, and then setting the "Read only" checkbox again immediately. If you turn off the "Read only" checkbox, change the text field, and then decide to move to another card and then come back, the text item will no longer be a "title field" and the contents will no longer be shared with all cards in the group.

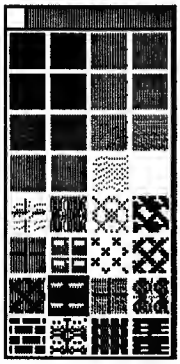
☒ **Hidden:** When the "Hidden" checkbox is marked, the entire text item is invisible. (It will show up, however, when you switch to the Edit tool or the Text tool.)

☒ **Transparent:** If this option is marked, the background of the text item will be transparent—the text will "float over" whatever is underneath it. This can be a very nice effect, especially if you set the text item's attributes so that HyperStudio does not draw a frame around the object or a scroll bar. Note that, if you do try to edit or scroll the text, the text item will immediately become opaque.

Add a Video... (Ⓒ-L): If you have a laserdisc player and an Apple II Video Overlay Card, HyperStudio will let you add a motion video sequence to a card almost as easily as you add clip-art. If you don't have the VOC, you can still attach a video sequence to a card; you'll just need a second monitor to view it. This procedure is identical to that of adding a video sequence to a button—the only difference is that the video will be added to the card itself, and will be played when the user first moves to that card.

If you select "Add a Video" from the Objects menu and a card video already exists on the current card, a dialog box will appear asking if you want to remove the card video or edit it.

For more information, see the description of playing a video in chapter 4, "Adding a Button," for details.



Colors menu

Selecting an item from this menu sets the current color for the paint tools. HyperStudio uses 16 standard dithered colors in the 640-Super Hi-Res mode of the Apple IIGS. There are 16 additional patterns available.

You can "tear off" this menu if you want to keep it handy on the screen—simply drag the mouse through the bottom or the sides. Drag it by the top bar to move it or click in the upper left corner (the close box) to make it go away.

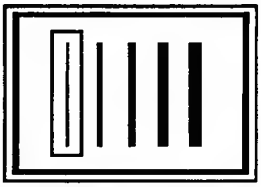
To edit a pattern, tear off the menu and then double-click the pattern you want to change. Any changes you make to the patterns are stored in HyperStudio, not your stack, if the HyperStudio program is unlocked. (You can lock and unlock HyperStudio by selecting its icon in the Finder and then picking "Icon Info" from the Special menu.) We expect that an Extra to load and save patterns will be available soon. (You can also edit the pattern by choosing "Edit Pattern..." in the Options menu.)

You can load pictures created with other paint programs that have custom colors, but you must be careful in using these graphics on a card. A custom color set may interfere with the appearance of the menu bar, or other buttons or graphics on the screen. Full-screen backgrounds with no menu bar or visible buttons work best for graphics with custom colors.

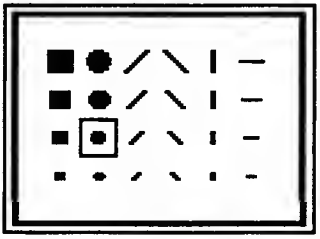
You can also set the current card to a custom color set, as long as the current card does *not* have the "Lock colors" attribute marked in the "Card Info" dialog. Just choose "Color Editor" from the Extras menu and click "Load Color Set." This allows you to load either a color set file or use the colors in any clip-art image. This will change the colors on that card. The `HS.Art` disk includes several files, including `Colors.Grays` and `Colors.Earth`, that can be used this way to set the palette for a card.

Options menu

This menu is used to control the operation of other functions in HyperStudio, particularly the paint tools. Many of commands in this menu may be summoned by double-clicking a paint tool—see appendix A, "Tool Reference," for more information.



Line Size...: This sets the width of the line used by the Line, Rectangle, Rounded Rectangle, Oval, and Polygon tools. When you choose this item, you'll get a dialog box that shows different line sizes—click the one you want.



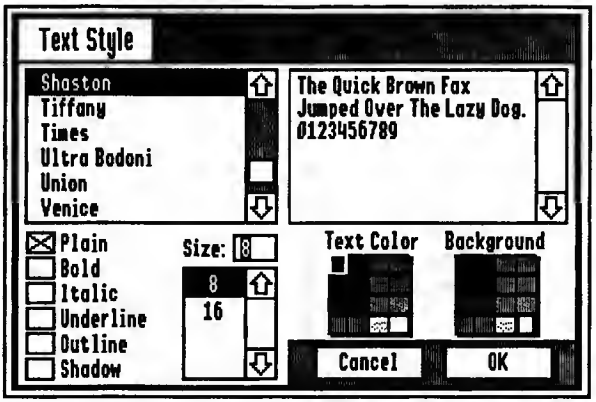
Brush Shape...: With this command, you may select the brush shape of the Paintbrush tool. You'll get a dialog box that shows the different brush shapes available—click the one you want.

Draw Filled: Choosing this item places or removes a check mark in the menu. This menu option indicates whether or not shapes drawn with the Rectangle, Rounded Rectangle, Oval, Freehand Shape, and Polygon tools should be filled with the current color or pattern. If "Draw Filled" has a checkmark next to it, the tools will draw a solid shape. (Select this command from the menu to toggle the checkmark.)

Draw Multiple: Choosing this item places or removes a check mark in the menu. If "Draw Multiple" is checked, the Rectangle, Oval, Polygon and Line tools will draw multiple images as you drag the mouse. The result is an interesting paint effect. (Select this command from the menu to toggle the checkmark.)

Draw Centered: Choosing this item places or removes a check mark in the menu. If checked, the Rectangle, Oval, Polygon and Line tools will draw from the center instead of a corner (the Rectangle, Oval, or Polygon tool) or an endpoint (the Line tool).

Text Style...: By selecting this command, you are brought to the Text Style dialog box.



Here, you may select a font and its size, any styles you would like applied to the font (boldfaced, italicized, underlined, shadowed, or outlined), and the text color. (A sample of the text as you have selected it will appear in the upper right corner of the dialog box.)

If you have selected a text item before choosing "Text Style...," you may also specify foreground and background colors, and all changes that you make will be made to the text item that you selected. Otherwise, the text style that you select will be used for the Text paint tool.

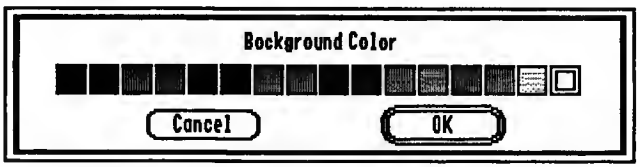
Listed are the fonts that you have installed on your boot disk in the Fonts folder.

Note that when you click in a text field, select a passage of text, or use the pointer or text tools to select an entire graphic object, selecting "Text Style" will show you what font, style, size, etc. the selected text is in. This can be useful to quickly see what style was used somewhere in a stack. (Identification, however, does require that the font be available on your boot disk.)

However, what if you wanted to assign a set of style settings from one text field to another? (That is, if you wanted to copy-and-paste the text font, styles [such as bold, italic, etc.] and the color, but not the text itself.) Here's an easy way:

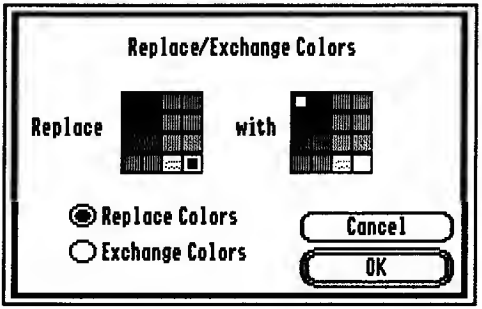
Select the first text field, and choose "Text Style" from the Options menu (or just press Apple-Y). When the style dialog appears, verify the styles visually, and then choose "Cancel." Now, the trick: select a new text field, but this time press Shift-Apple-Y (or hold down the shift key while you select "Text Style" from the Options menu). The old style will be carried into the style dialog. Clicking "OK" at this point will then set the second text field to the style of the first. This can be very useful if you have to change the style settings of several different text fields.

Text Color...: This command sets the color used by the the Text paint tool, or, if you have a range of text selected in a text item, the color of that text. (If you have an entire text item selected, all of the text will change to the color you specify.)



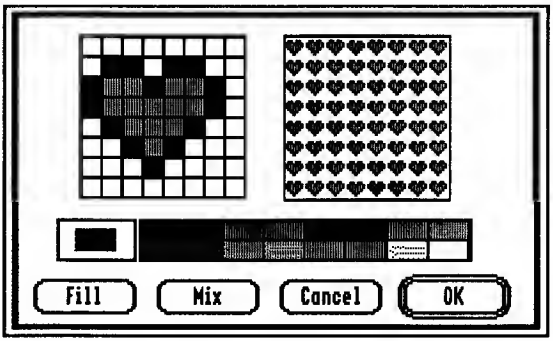
Background Color...: This command is used to set the background color (or pattern), used by the "Erase background" command (in the Edit menu), and by the Eraser tool. Additionally, if you have a range of text or an entire text item selected, this command will change its background color. When you select this command, will be presented with a dialog box containing all of the colors and patterns in HyperStudio; simply click one to use it. (If you are using a paint tool, the dialog box will also contain

an "Erase Now" button, which you may click if you want to immediately erase the background.)



Replace Colors...: This command allows you to replace one color with another (in a selected area or over the entire background), or to exchange (swap) one color with another. To use it, first select the area that you would like to work with by using the selector tool or the lasso, or don't select anything to work with the entire background.

Next, select "Replace Colors..." from the Options menu and click the action that you would like to perform (replace or exchange), then select the two colors from the palettes. Clicking "OK" will perform the action; if you don't like the results, choose "Undo" from the Edit menu immediately.



Edit Pattern...: After you have selected a pattern from the Colors menu, this command is available and will display the pattern editor dialog box, which allows you to edit the current pattern. You will be presented with a picture of your pattern in "fat bits" mode on the left side of the dialog box, and on the right side, a rectangle painted in your pattern (so that you may get an idea of how it looks when repeated).

In the bottom of the dialog box, you may select the color that you wish to edit with, and you have four options: "Fill" (to fill the pattern completely with the current color), "Mix" (to put the current color in every other pixel of your pattern), "Cancel" (to exit without saving any changes), and "OK" (to keep your new edited pattern).

Patterns are stored in HyperStudio, not your stack, if the HyperStudio program is unlocked. (You can lock and unlock HyperStudio by selecting its icon in the Finder and then picking "Icon Info" from the Special menu.) We expect that an Extra to load and save patterns will be available soon.

Hide Items: This command simply causes all buttons, graphic objects, and text items to become invisible and inactive. You may want to use this to save a picture of the background without any objects showing (with "Save Screen...," in the File menu) or to paint on the background without any objects in the way. Note that buttons can't be clicked when they're hidden—this command effectively makes them disappear. Choose this option a second time to unhide all of the items.

Hide Menu: This will hide the menu bar on a single card. If the menu bar is hidden when you move off a card, it will also be hidden when the user views that card. This allows you to create stacks without a menu bar. Press Apple-M to restore (or hide) the

menu bar. To hide or show the menu bar on all cards in a stack, you should use the Menu Tamer extra, explained in appendix D, "Sample NBAs, Extras, and Transitions."

chapter

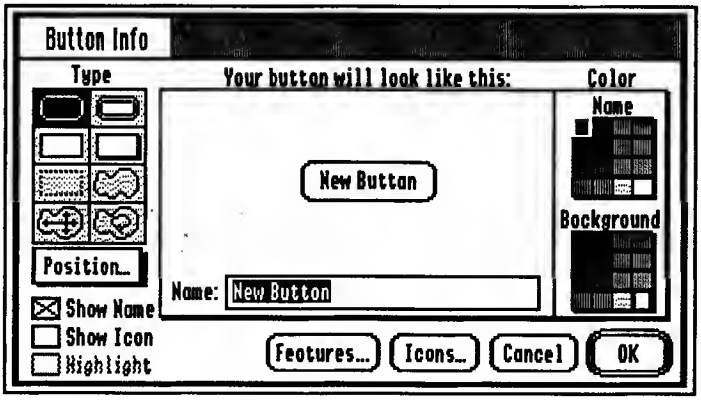
4

Adding a Button

This chapter sheds light on the cornerstone of HyperStudio, buttons, by explaining the button addition process, step-by-step.

Adding a Button

Add a Button... (⌘-B): Selecting this option allows you to add a button to the current card. After choosing this option from the menu, you'll be presented with the "Button Info" dialog box:



the Button Info dialog

On the right side of the "Button Info" dialog box are the eight different button types that you may create. Below the eight button types is a "Position" button (allowing you to position the button on the card), and several button options ("Show Name," "Show Icon," and "Highlight").

In the middle of the dialog box, a sample of your button will be shown, and immediately below that, an edit line, where you may type in the name of your button. On the far right of the dialog box are the color palettes, where you may set the foreground and background colors for your button.

Finally, across the bottom of the dialog box are a number of buttons: "Features..." (where you can control button features such as whether or not it's a group item), "Icons..." (where you may select an icon to be associated with your button), "Cancel," and "OK."

☞ Novice User

You won't see the "Highlight" checkbox or the "Features..." button. These are both Advanced User features.

The Eight Button Types

There are eight different types of buttons that you may define in HyperStudio. The first four are visible and automatically highlight when clicked, and the remaining four are invisible and will highlight when clicked only if you have the "Highlight" checkbox marked (which appears when Advanced User mode is turned on, and is automatic for novice users).


Rounded Rectangle. The Rounded Rectangle button is the default and the simplest. It simply looks like a rectangle with rounded corners.

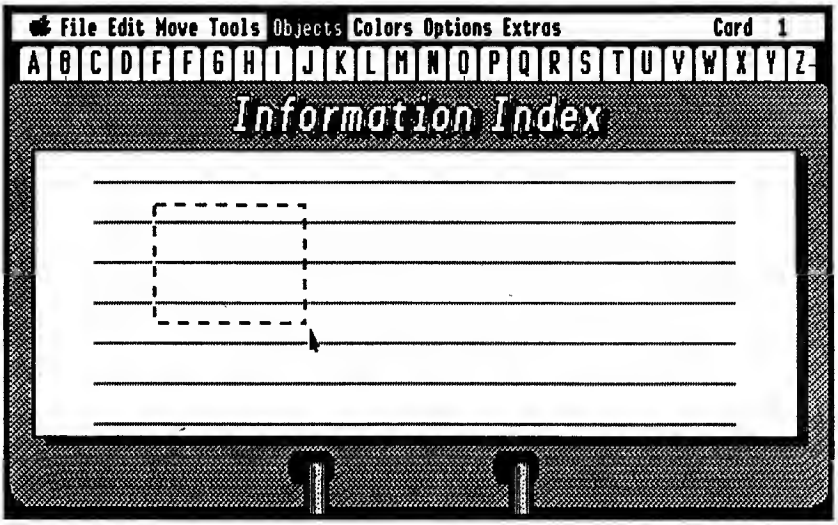

Rectangle. The Rectangle button is also a very simple button. It looks like a rectangle.


Bold Rounded Rectangle. The Bold Rounded Rectangle button is similar in appearance the Rounded Rectangle button, except that it has an additional outline around the outside of the button for emphasis. One other important difference is that, on a card with a Bold Rounded Rectangle button, pressing Return will trigger the button. Therefore, you may not have a text item on the same card, if you wish the user to be able to enter text and press Return.


Shadow. A Shadow button is the same as a Rectangle button, except that it has a small drop shadow behind the button.

The following four button types are all invisible. This means that they still act as buttons (clicking them causes an action), but don't show up on the screen.


Rectangular Invisible Button. This is simply an invisible version of the Rectangle button, above. The button name or any icon associated with the button will not show up, however, unless you check the "Show Name" and/or "Show Icon" checkboxes. After selecting this button type and clicking "Position...", you are shown a rectangle with a dashed border, representing your button.



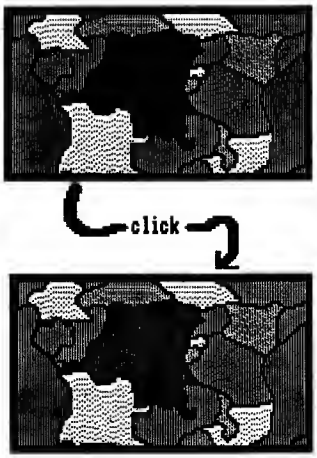
placing a rectangular invisible button

Drag inside of the rectangle to position the button, and drag from the rectangle's corners to resize the button. Click outside of the rectangle or press Return when you have it positions as you like.

Freehand Area Button. This button type allows you to draw the shape of your button, freehand. The button name or any icon associated with the button will show up on the button if you check the "Show Name" and/or "Show Icon" checkboxes. After selecting this button type and clicking "Position...", you are presented with your card and a pencil cursor. Draw the button as you want it and when you are done, HyperStudio will prompt you to "Try Again" or accept what you've drawn (by clicking "OK").

Expanding Area Button.

This button type allows you to specify the inside of your button area, and it will determine the appropriate edges. The button name or any icon associated with the button will show up on the button if you check the "Show Name" and/or "Show Icon" checkboxes. After selecting this button type and clicking "Position...", you are presented with your card and a four-headed arrow cursor. Click inside the boundary of your button, and HyperStudio will work out from that point to find the edges of the button. (This is very useful when you have a map, for example. You can click in the center of a state, and HyperStudio will go out from that point to find the edges of the state.)



After the shape of your button is determined, HyperStudio will show you the button and prompt you to "Try Again" or accept what it's created (by clicking "OK").

Lasso Area Button. This button type is the opposite of the Expanding Area Button, above: it allows you to lasso an object on the screen, and HyperStudio will hug up against the object to define the button's shape. The button name or any icon associated with the button will show up on the button if you check the "Show Name" and/or "Show Icon" checkboxes. After selecting this button type and clicking "Position...", you are presented with your card and a lasso cursor. Simply use the lasso and draw around the object that will define the button's shape. If the object is on a solid background, you don't need to drag neatly; the lasso will contract and shrink down to find the edges of the object.

After the boundary of your button is determined, HyperStudio will show you the button outline and prompt you to "Try Again," or accept what it's created (by clicking "OK").

Button Attributes

☒ **Show Name:** Marking this checkbox makes the button name visible—whatever is typed into the name box will be shown inside the button in 8-point Shaston, in whatever color is selected in the "Name" color selection palette. Button names that describe an action ("Next Card", "Cancel", "Launch BongoWorks") can help the user. Button names can stand for the response from the user ("True", "1492", "All the above"). You do not *have* to name the button, although it may be useful. Scripts and stacks that use the test functions may refer to the button's name.

If "Show Name" is checked, the button name will appear in whatever color is selected in the "Name" color selection palette. The rest of the button will be the color chosen in the "Background" color selection palette.

☒ **Show Icon:** Marking "Show Icon" will show the icon associated with a button, if any. You can select an icon for a button by clicking the "Icons..." button (described below).

✔ **Advanced User**

☒ **Highlight:** Some buttons reverse their colors when they are clicked. This is called highlighting. The first four button types (the visible buttons) always highlight. With this checkbox, you can cause a button of one of the last four types (the invisible buttons) to highlight when clicked.

Highlighting is always on for novice users.

Button Options

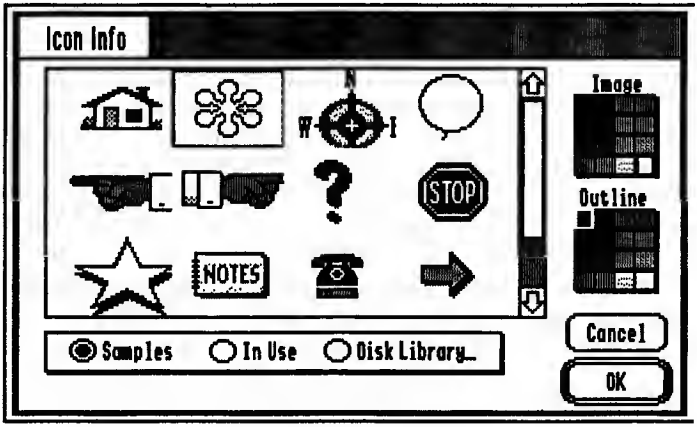
Icons...: Clicking this button opens the "Icon Info" dialog. In this dialog box, you may select an icon to be associated with a button, and give the icon a color.

To select an icon, simply click it. HyperStudio defaults to showing you the icons built into the program; you can click "In Use" to look at icons already in use in the stack, or "Disk Library..." to select a Finder icon file from disk.

✔ **Advanced User**

Clicking "Disk Library..." not only allows you to select a Finder icon file, but will allow you to open any file that may have embedded icons. After opening that file, if the file contains any icons, HyperStudio will allow you to use them. Note that many HyperCard stacks contain icons.

If an icon's color scheme may be changed, the two color palettes on the right side of the dialog box (for the icon image and outline) will change from gray to the current range of colors. Simply click the colors you wish for the icon and its outline.



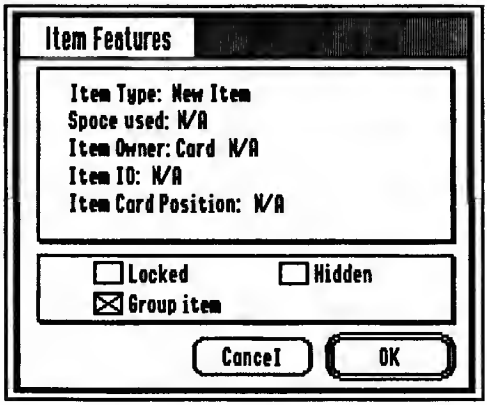
the Icon Info dialog box

✔ Advanced User

Features...: This button is only available if you are an Advanced User.

Features...: Clicking "Features..." opens a dialog box which allows you to control the button's features.

This dialog box (which is the same as the "Button Info..." dialog) tells you a little bit about your button and allows you to change some of its features. Since the button hasn't been created yet, the upper portion of the dialog doesn't hold much useful information.



However, the lower section contains three checkboxes: "Locked," "Hidden," and "Group item."

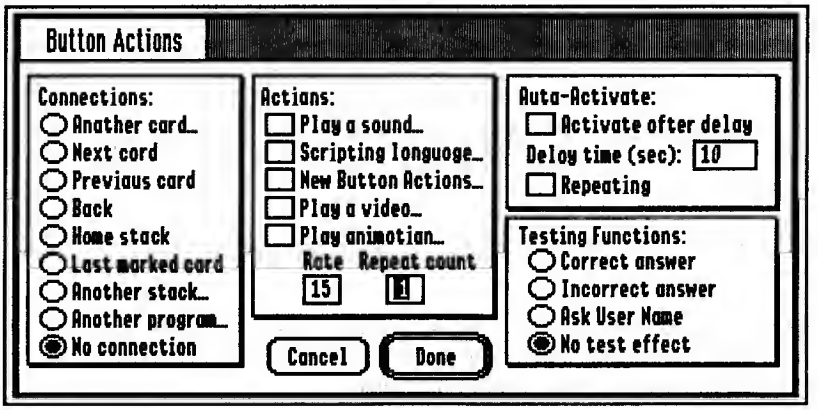
☒ **Locked:** If a button is locked, it can not be moved or deleted.

☒ **Hidden:** If a button is hidden, it will not appear on the screen or be active. (That is, a hidden invisible button will not respond to clicks.) You can unhide a button by simply unmarking this box in the "Button Info..." dialog box, or through the scripting language.

☒ **Group item:** If a button resides on a group card, it can be made a group item. This means that it will be shared across all other cards in the group.

Button Actions

After you position the button on the card, you may specify which button actions will occur. The Button Actions dialog box will appear:



the Button Actions dialog box

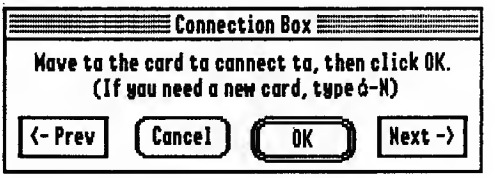
Novice User

You won't see the far right of this dialog box (the "Automation" and "Testing Functions" sections).

Connections

This section of the Button Actions dialog allows you to specify where to go when the button is activated. The choices are:

- **Another card...:** This connects the button to another card in the same stack or another stack. When you select this, you'll be temporarily returned to the current card, and the Connection Box window will appear.



  You may use any of the usual commands in the Move menu to navigate to another card in the current stack. (Or, you may use the "Prev" and "Next" buttons in the Connection Box window.) Additionally, if you want to connect to a card in another stack, simply open the new stack, and move to the card you wish to connect to. In either case, when you find the card you want to connect to, click "OK". HyperStudio will create a "hard link" to the other card. (This is in contrast to a "soft link" such as "Next card" and "Previous card," below.)

- **Next card:** This will link the button to the next card in the stack. Note that HyperStudio stores this internally as a "soft link;" instead of keeping the actual card ID of the next card, it simply knows to go to the card after the current one, when this button is clicked. As a result, you can copy and paste this button throughout a stack and it will always go to the next card.

- **Previous card:** Like "Next card," above, this simply instructs the button to go to the previous card. Also like "Next card," it is a soft link, which allows you to use the same

"Previous card" button throughout a stack, if you like. Remember that a stack forms an imaginary ring of cards; if you choose "Previous card" from the first card in a stack, HyperStudio will take you to the last card in the stack.

- **Back:** This connect brings the user to the last card they visited—even it it's in another stack. This "back list" (or "recent list") contains the most recent 16 cards that the user visited.

- **Home stack:** This creates a link to the last stack that the user loaded named `Home.Stack`.

- **Last marked card:** The "Card Info..." dialog box (in the Objects menu) allows you to mark a card. Choosing this button connection will link the user to the last marked card they visited, even if it was in another stack.

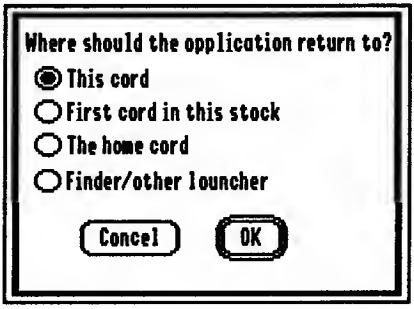
- **Another stack...:** Selecting this connects the button to another stack that you specify. After selecting "Another stack...", the standard file dialog will appear. You may choose a stack in another directory, or even on another disk. (If the disk or directory can't be found when the button is pressed, the user will be prompted to first insert the missing disk, and if that doesn't work, to locate the files "by hand" with the standard file dialog.)

  HyperStudio always looks in the current directory for a stack, graphic, sound, application or any other disk-based item before looking at the location specified by the full pathname. The only exception to this rule is that you may have a file named `Home.Stack` in another directory and HyperStudio will link to it. Other than that, however, you may not have a stack in your current stack's directory which has the same name as the stack you are linking to in another directory—HyperStudio will simply see the stack in the local directory before it looks out in another directory!

  You can have stacks which access information in subdirectories, and you may move them, as a whole, to other locations. HyperStudio will remember the hierarchy and the stacks will still function.

  Note that "Another stack..." is used to connect to the *first* card in another stack. If you want to connect to another card in the stack that isn't first, use the "Another card..." option, explained above.

- **Another program...:** A button can connect to another application (such as AppleWorks, the Finder, `BASIC.System`, or any other program). After clicking "Another program...", you will be prompted (via the standard file dialog) for the program that this button should run. After you have selected the application, a dialog box will appear, asking you where the application should return after it quits. You can choose to return to the card that contains the button which launched the application, or to the



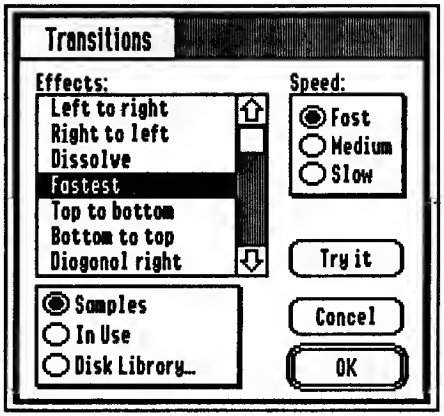
first card in the current stack, the home card, or the program that originally launched HyperStudio.

- **No connection:** Choose this connection if you want the user to remain on the current card. You may pick this if you want a button to simply play a sound or trigger a video, for example, without any other concurrent actions.

Transitions

All of the button connections (except "Another program..." and "No connection") give you the choice of a visual effect, also known as a *transition*. The transition makes the movement from one card to another more visually interesting.

After choosing a connection, the Transitions dialog box will appear. It allows you to select a transition and a speed, and to try the transition out (via the "Try it" button). Note that you can specify where you wish to look for transitions: from the samples included in HyperStudio, from those transitions already in use, or from a transition library on the disk.

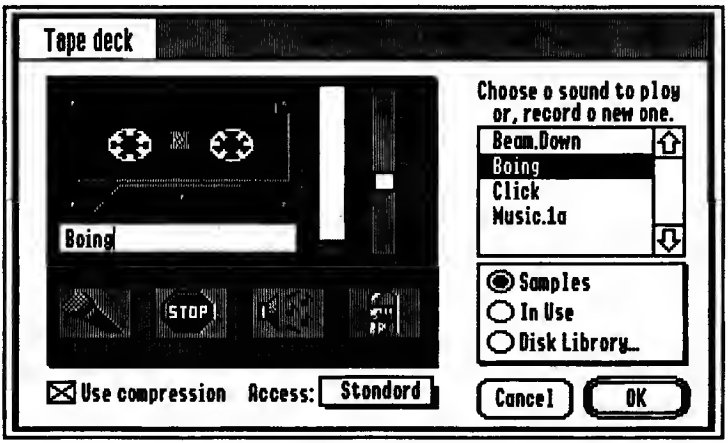


Keep in mind that the medium and slow transitions are processor-speed independent: that is, they will always execute at the speed you specify (medium or slow), regardless of the speed of the computer. The "fast" transition speed will vary with the Apple IIGS' speed and will always run as fast as possible.

Other Button Actions

Other actions besides (or in addition to) branching to another card, stack or application are possible. These are discussed in detail in the following pages.

☒ **Play a Sound...:** You can have a sound play when a button is pressed. If you choose this action, you will see the Tape deck dialog box. It will allow you to select an existing sound or record a new one.



the HyperStudio Tape deck

☞ Novice User

You won't see the "Use compression" checkbox or "Access" pop-up menu, at the bottom of the Tape deck. They are advanced user features.

If you'd like this button to play a sound that already exists, simply click the sound's name in the sound list, in the upper-right corner of the Tape deck. Note that you may choose where you want to use a sound from: the samples included in HyperStudio, the sounds currently in use, or from a disk file.

✔ Advanced User

In addition to being able to select standard sound files, the "Disk Library..." option will allow you to select any file that may have an embedded sound. If that file has sounds, they will be listed in the sound list. Frequently, applications and HyperCard stacks will contain sounds.

Click the Play button to verify the sound you want to add, and that the volume is at an acceptable level. To use the sound, click the "OK" button.

If you wish to record a new sound, first speak into the microphone and verify that the level indicator is moving. If it turns red, the level is too high (the sound is too loud) and if the level indicator barely moves, then the sound is too soft.

When you have the volume set as you like, click "Record." The screen border will turn green, indicating that HyperStudio will begin recording as soon as it hears a sound. When you begin talking (or you otherwise start your sound input), the border will turn red to show that HyperStudio is recording. Press any key to stop recording.

Tips for recording a good sound: you need to hold the microphone close to your mouth, almost resting on your chin. Speak firmly and clearly into the mike; try to use a "radio voice" and enunciate.

Click "Play" to hear your sound, and if you are not satisfied with the recording, simply record it again. When you are done, type a name for the new sound in the edit line (below the picture of the cassette tape) and click "OK". The sound will automatically be saved in the stack.

✔ **Advanced User**

You have several options for saving your sound. The first is a checkbox, "Use compression." If you mark this, your sound will be compressed before it is stored in the stack, and when the button is clicked, quickly uncompressed and played. The quality of your sound decreases very slightly but the resulting space savings may be worth it.

The remaining options, in the "Access" pop-up menu, control where the sound is stored.

The first option is "Fastest," and places the sound in the stack with the button definition. This will take up the most amount of memory, but is also the quickest to retrieve a sound from memory. It is easy to make a copy of the stack since everything is in the same place, however, sounds stored in this format may not be shared with other buttons. (This method is sometimes called an *embedded* sound.)

The second option is "Standard." This stores your sound in your stack in a location where it can be shared with other buttons. As a result, any sounds stored in this manner can be used by other buttons in the stack. Also, large stacks or those that take up a lot of memory will run more efficiently because HyperStudio will not load any sounds from the stack unless it needs them, and then, if it needs the memory for something else, HyperStudio will remove the sounds from memory (since it can always pull them from the stack again).

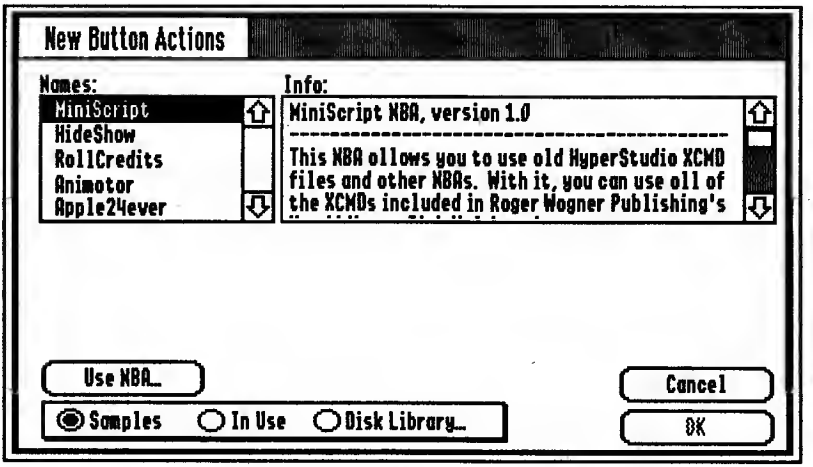
The final option is to save the sound to a "Disk file." This is an efficient way to store a sound library, and sounds may be shared by many stacks. However, this is also the slowest option (since HyperStudio must load the sound from disk every time it is to be played). Also, if you give a copy of your stack to somebody, you have to remember to copy the sound file(s), as well.

If you have a sound file that was not created with Sound Shop or HyperStudio and you wish to use it in a stack, simply load it into Sound Shop and then save it as a HyperStudio-format sound. Note that most other sound file formats do not contain information about the playback rate and volume settings, so you may need to adjust those settings in Sound Shop before saving the edited sound to disk.

☒ **Scripting Language...:** Selecting this allows you to access any programming languages built into HyperStudio. (HyperStudio ships with the SimpleScript language, but has the capability to accept other languages.) If only one language is built into HyperStudio, you will automatically be enter it, otherwise, you will be given a list of languages available and prompted to choose one. For more information, see chapter 5, "SimpleScript."

☒ **New Button Actions...:** This command allows you to access the New Button Actions (NBAs) in HyperStudio. As the title indicates, a New Button Action is HyperStudio's way of providing endless possibilities in future expansion of the system. NBAs give you a way to continually add new actions to the list of options for a button. HyperStudio comes with several sample NBAs.

Clicking the "New Button Actions..." checkbox will bring up the New Button Actions dialog box:



the New Button Actions dialog box

In the upper-left corner of this dialog is a list of all of the sample NBAs, built into HyperStudio. (You can change to a list of NBAs already in use or residing in a disk file by clicking "In Use" or "Disk Library...", respectively, at the bottom of the dialog.)

Immediately to the right of the list of NBAs are instructions for the current NBA. It will normally contain information on the name of the NBA, the author, and information on how to use it. Additionally, if the NBA can work with SimpleScript (HyperStudio's scripting language), that information will appear there, as well.

If you would like to use the NBA, click the "Use NBA..." button, at the bottom of the screen. This will pass control to the NBA, and if the NBA needs any additional information from you, it will prompt you as appropriate. (For example, an NBA that dials a phone number stored in a text item may prompt you for the text item from which to get the phone number.)

For more information on the NBAs included with HyperStudio, see appendix D, "Sample NBAs, Extras, and Transitions."

☒ **Play a video...:** This option allows HyperStudio to start a video sequence on a Pioneer 2200, 4200, or 8000 laserdisc player attached to the Apple IIGS via the modem port or printer port. (The "Play a video..." command can not control a VCR or other video source, however.) Additionally, if you have an Apple II Video Overlay Card, HyperStudio will control it and permit you to specify a rectangle where you wish the video to appear.

You may also enter a laserdisc video sequence without a laserdisc player being attached to the computer; this is useful if you want to set up laserdisc stacks at home, for example, for use at school (where the laserdisc player is located).

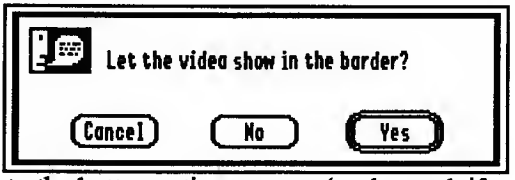
Whenever the button is pressed, the video sequence will be shown once. (If you want it to run automatically after a set period of time or repeat the sequence, see the section on auto-activate buttons, below.) A video sequence can be attached to a card so the video is shown as soon as the user moves to that card. For more information on that feature of HyperStudio, see the "Add a Video" section in chapter 3, "Reference."

To attach the laserdisc player to your computer, use a "CC-04" cable with the proper connectors on each end. In the Apple IIGS Control Panel, go to "Slots" and be sure that slot 2 is set to "Modem," and then, in the "Modem" control panel, be certain that the "Baud Rate" setting is 4800.

When you click "Play a video...", the computer will check for a Video Overlay Card. If it doesn't find one, it will immediately begin searching for a laserdisc player. Please skip to "Using the Laserdisc Remote Control," below.

If you do have a Video Overlay Card, HyperStudio will ask if you wish to use live or laserdisc video. Live video is video that you provide when the button is clicked (or, in the case of a card video, whenever the user moves to the card).

After choosing live or laserdisc video, the computer will then ask if you want to show the video in the screen border. This simply controls whether or not the video can fill the entire screen.



If you chose to use live video, you are done defining the sequence and will return to the button actions screen (or the card, if you are adding a card video). You may then use the paint tools on the card to paint in the key color (discussed below) and create an area for the video to show through.

If you specified that you wish to use laserdisc video, HyperStudio will display a rectangle in the key color. This rectangle determines exactly what portion of your video image will be displayed on the card. You can position the rectangle by dragging the edges, or resize the rectangle by using the handles on the corners. Note that changing the size of the rectangle does not shrink the entire video image into the new rectangle, but simply clips it at the boundaries.

Using the Laserdisc Remote Control

At this point, HyperStudio will search for a laserdisc player attached to the modem port. If it is unable to find one, it will request that you check the connections and ask if it should "Try Again," "Continue," or "Cancel." Clicking "Try Again" will cause the computer to look for the laserdisc player again, while "Cancel" will cancel the entire process.

If you click "Continue," *or* if a laserdisc player is found, HyperStudio will put the laserdisc "remote control" at the bottom of the screen.

the laserdisc "remote control"

A video sequence is defined by the beginning and ending frame numbers. The radio button starts on the "Begin" setting. Click the "Play" or "Step" buttons to find the beginning of the sequence you want. (Note that, if you're editing a video sequence previously defined, the beginning and ending frame numbers will be pre-set, as well as the other information stored about a video, such as the playback speed and direction.)

While the laserdisc is playing, click "Step" or "Pause" to hold the disc on the current frame. (When you do either of these, the frame counter to the right of the "Begin" button will update.) You can also press the Clear key and enter a new frame number manually. The player will jump to that frame.

To set the ending frame, click the "End" radio button, then click "Play" to let your video roll by before pressing "Stop" or "Pause." If you want to show only a single frame of video, make the ending frame number the same as the beginning. You can use the TAB key to switch between the frame counter to the right of the "Begin" and "End" buttons. When you enter a frame number in a box, the radio button will automatically move to that frame.

If you have a laserdisc player attached to the Apple IIGS, you can test your video sequence by clicking the "Test" button. When you get the sequence you want, click the "Keep" button.

If you don't have a laserdisc player, you may still manually enter the beginning and ending frame numbers for the sequence you wish to define, then click "Keep" to store them in the stack. They will play back, as normal, when you are using HyperStudio on a computer that does have a laserdisc player attached.

Other features in the remote control that you should be aware of:

Play: You can play forward or backward by clicking on the arrow buttons to either side of the word "Play." If the end frame number is less than the beginning frame number, the sequence will be played backward.

Step: Click either side of the word "Step" to move one frame forward or backward.

Playback Speed: Click the buttons to the left and right of the word "Norm" to change the play speed of the video. Faster speeds include 2X (double speed) and 4X (quadruple speed). Slower speeds range from ½ speed down to 1/16 speed. There are even two special "slide show" modes: they show one frame every 1 or 2 seconds. Note that the speed seen by the user will be whatever speed is there when you click "Keep." That means you can use the faster speed settings when looking for the frames you want, and then set it back to "Norm" (or whatever you desire) before clicking "Keep."

Sound: By clicking on the button normally marked "Stereo," you can switch the sound off ("silent"), or select either the left or right audio channels. Sound is not available at playback speeds other than "Norm."

Frame: If you click the button marked "Frame" (on CAV discs; CLV discs display "Time") to the right of the frame number indicators, the button will change to "Chapter" (for Chapter). The number display will show what chapter that frame number is found in. If you click the chapter number while in the chapter mode, the laserdisc player will seek to the beginning of that chapter. When in the chapter mode, you enter two digit numbers, and in the frame and time modes, you enter five digit numbers.

Display: Clicking the "Display" button will toggle the current frame number being displayed in the upper-left corner of the screen. Note that this frame number comes from the laserdisc player and is part of the video signal; if you don't have that part of the video showing through in that corner of the screen, you won't see the frame number. Frequently, it will be hidden under the menu bar; you have to press Apple-M to turn the menu bar off before pressing Apple-L to add laserdisc video.

Window: Toggling this checkbox will cause the video rectangle to appear or disappear.

Start/Reject: If the laserdisc player is not fully started when you bring up the remote control, the button in the upper-left corner of the box will read "Start." Click it to start the player. (An indicator that the player must be started is that the remainder of the controls in the remote control will be dimmed.) Once the player starts, this button changes to "Reject." Clicking it will eject the laserdisc.

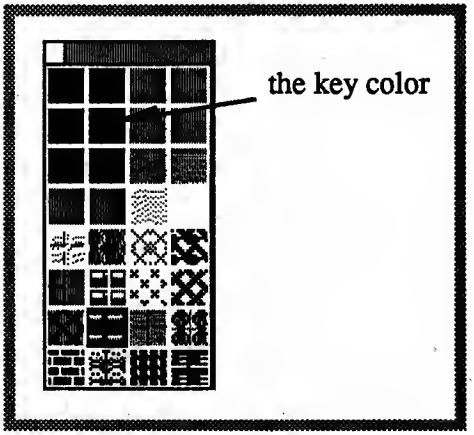
Cancel: Clicking this tells HyperStudio that you want to back out of the current operation. If you were editing a video sequence, this will make no changes to the sequence. Otherwise, this will tell HyperStudio that you didn't wish to make a new video sequence.

(Note that what HyperStudio stores in your stack is the starting and ending frame numbers of the video sequence, not the actual video images. Your computer must be attached to a laserdisc player in order to view the sequence again.)

Using Full-Screen Video and the Key Color

In normal use, the video image (with the Apple II Video Overlay Card) will only show through the area defined by the video rectangle used when Adding a Card Video. However, the Video Overlay Card works by defining a color as the "key color," or "show-through color," on the screen. Before a video is added to a card, the key color is purple. The purple hue on the screen changes to a pure blue once a video object has been added to a card.

There are two degrees of "full screen" on the Apple IIGS. The first is the active screen area, excluding the border area. The larger full-screen includes the monitor border area as well. You can design a card to use either method.



If you want to create a video image that fills the background of a card, paint the entire card background with the key color. The color to choose for filling the background will either be purple or blue, depending on whether a video has already been added to a given card. Its position in the HyperStudio color table will always be the same: the sixth color in the "Colors" menu, which is the second color from the left in the second row from the top. If you are using the "Background Color..." menu option (Options menu), use the 6th color choice from the left.

When your video image is displayed, the video will only show where you've painted in the key color. (To get rid of the video rectangle, unmark the "Window" checkbox in the laserdisc remote control.)

If you want the video image to fill the screen border area as well, click "Yes" when you see the dialog box asking if you want to show the video in the screen border area (when first adding a video). The only difference between these is that HyperStudio changes the Apple IIGS border color to the "key color." This will allow the video image to go to the absolute edge of the monitor screen.

The advantage of using the background graphic itself (as opposed to the video rectangle) for the video show-through area is that it is easy to repaint a rectangle or other area if you want to change the image area. You can use the "Add Clip Art..." function or other paint tools to overlay labels, arrows, or icons onto your video image.

Try drawing some solid rectangles (with the "Draw Filled" checked in the Options menu) in various colors. You'll find that, since the 16 colors used in HyperStudio are actually combinations of other colors, your rectangles will range from completely transparent (blue) to completely opaque (black, white, gray, and shades of red). Others are in between. This can create neat effects that show a dimmed image of a larger video image, with a certain part "highlighted" (brightest).

The key here is to experiment. You can't hurt anything! You're likely to discover many useful techniques in the process. Try the paint tools to create small windows with video

showing through. We've found that the video rectangle is the easiest to explain and use for beginning users, but using the paint tools gives the most flexible results.

Turning off Live Video

Essentially, clicking a button (or going to a card) containing a "live" video simply enables the Video Overlay Card and then returns control to HyperStudio with the VOC on.

The Video Overlay Card is automatically turned off when HyperStudio moves to another card in a stack, so, the easiest way to turn the VOC off is to simply create a button that goes to another card which doesn't have live video.

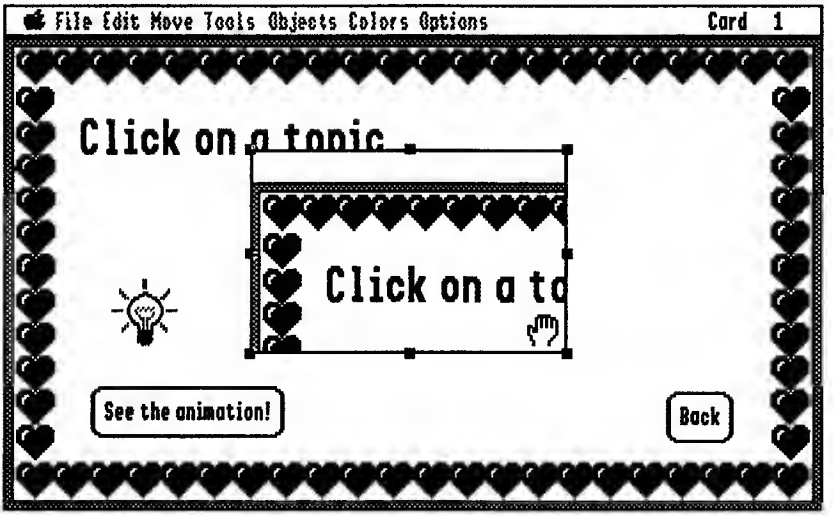
☒ **Play Animation...:** This option allows you to trigger an animated sequence when the button is clicked. HyperStudio supports the "PaintWorks" style of animation, used in PaintWorks Plus, PaintWorks Gold, and Platinum Paint. This is animation in the traditional sense, where a number of still images are played back rapidly to create the illusion of movement.

PaintWorks stores its animation files on the disk with a filetype of $C2, and you may hear them occasionally referred to as "$C2 animation files." HyperStudio has the ability to create these files (internally) on the fly, so that you can make new animations in HyperStudio without having to own any other program.

To add an animation to a button, click "Activate animation..." HyperStudio will present you with the standard file dialog, and you may select either a $C2 animation file or a super hi-res picture.

If you select a super hi-res picture, HyperStudio will assume that you wish to build an animation from a series of pictures stored on disk, that end with consecutive numbers. For example, if you select a file named `Bird.1`, HyperStudio will assume that you want to build an animation from the frames stored in the files `Bird.1`, `Bird.2`, `Bird.3`, and so on.

In either case, after selecting a file, you will be returned to the card, and the first frame of your animation will be in a window on the card.



adding an animation

Only the part of the graphic visible in the window you create is displayed during the animation sequence; if anything in your animation moves outside this area, it will not be visible. To position the animation *within* the window, drag the mouse around inside of the window. To position the window on the card (which controls exactly where on the card the animation will show), drag the frame of the window. Finally, drag the handles on the corners of the window to change its size.

Click outside the rectangle or press Return when you have everything in place as you like it. You will be returned to the Button Actions dialog box. (If the graphic you specified is part of a numbered series of frames, HyperStudio will generate the animation before returning to the Button Actions dialog box.)

The final step is to enter a Rate and Repeat count in the appropriate boxes on the Button Actions dialog. The rate is counted in 60ths of a second, and controls how fast the animation runs. A value of 30 would tell HyperStudio to show two frames per second. For general purposes, you can consider the rate values of 5, 15, and 30 to be similar to fast, medium, and slow, respectively. The "Repeat" value controls how many times the sequence repeats itself—enter "1" to have the animation play just once.

☞ Novice User

The following button action (Auto-Activate Buttons) is only available to advanced users.

Auto-Activate Buttons

Normally, a button must be clicked to be activated. An auto-activated button activates automatically (as if someone had clicked it) after a specified number of seconds. The button can also be set to continuously cycle.

Auto-activate buttons work well for a help system; a help card or item could be displayed if they user doesn't click something on a card in a certain amount of time, for example. Auto-activate buttons can also be used to create timed tests.

Note that auto-activate buttons are not active until you turn on the auto-activate function in the Preferences dialog box, *and* you are in the Browse mode. This way, buttons won't be activating while you're using an Edit or Paint tool and trying to work on your stack.

A series of auto-activate buttons can be used to "orchestrate" a sequence on a card. For example, you could use one auto-activate buttons to play a sound (after one second), then a second auto-activate button to start an automation after three seconds, followed several seconds later (via a third auto-activated button) with another sound.

☒ **Activate after delay:** click this box to cause a button to set up an auto-activated button, then enter a time (in seconds) after which the button should activate.

☒ **Repeat:** marking this box will cause the button to continuously re-activate after it has been triggered.

☞ **Novice User**

The following button action (Testing Functions) is only available to advanced users.

Testing Functions

HyperStudio's testing functions allow you to create a stack which keeps track of buttons that have been pressed, and stores the results in a text file on disk. Buttons may have one of four test effects: "No test effect," "Correct Answer," "Ask User Name," or "Incorrect Answer." Normally the setting is "No test effect," and buttons created this way are 'neutral'—they have no effect on the test status of a stack or card.

If you choose to make *any* button in a stack either "Correct Answer" or "Incorrect Answer," that stack is considered a "test stack." When a test stack is exited (by launching another application, branching to another stack, or simply quitting HyperStudio), HyperStudio writes to a text file in the same directory as the stack, named `HS.Test.Results`. The information written includes the stack name, the time and date, a list of which buttons were chosen on each card, and a final "score" for that stack. Information is written to the `HS.Test.Results` file on a cumulative basis, so this file contains the results of all stacks in that directory that are used until the file `HS.Test.Results` is deleted. The `HS.Test.Results` file is written to only if there are test effect buttons in the stack.

◉ **No test effect:** This is the default value for a button, and causes no test effects to happen. HyperStudio will not record if this button is clicked.

◉ **Correct Answer:** This causes the button to register a "correct answer."

◉ **Incorrect Answer:** This causes the button to register an "incorrect answer."

Ask User Name: When this option is marked for a button, the user is prompted to type in their own name. This does not change the button name. Note that the user's name is stored in the `HS.Test.Results` file.

The test results for a given stack are calculated as follows: when the stack is terminated, HyperStudio scans the entire stack to see what buttons were pressed as the stack was being used. For each card, it writes out the card's ID number, and which "Correct Answer" or "Incorrect Answer" button on the card was clicked.

If no button on that card was clicked, or if the button clicked had "No test effect" (the default), then no entry is written for that card.

If a button on the card defined as either "Correct Answer" or "Incorrect Answer" is clicked, HyperStudio will record the button name as the chosen button for that card. Finally, HyperStudio will write the number of "Correct Answer" buttons chosen (no more than one per card) and the total number of cards with test effect buttons.

There can be as many "Correct Answer," "Incorrect Answer," or "No test effect" buttons on a card as you want; only the most recently pressed button with a "Correct Answer" or "Incorrect Answer" function will be counted.

You may want to experiment with different button combinations to see what works best for you. Possible options include:

- A simple multiple-choice test: Each card contains one question. Each possible answer button moves the user to the next card. The "Correct Answer" test effect is used for the right answer on each card; the remaining buttons are set for "Incorrect Answer."
- Include forward and backward arrow buttons on each card. Define them as "No test effect." Clicking these will not affect the "last button clicked" status on a card, so the user could click a choice, and then click the forward button. HyperStudio would still record the test effect button chosen, and would ignore the forward button.
- A timed test: Each screen includes an auto-activated (no test effect) button that moves to the next card after 25 seconds (for example). The user would have to choose one of the answer buttons before the time is up. The final `HS.Test.Results` file would record which button on each card had been chosen.
- User profiles: A stack has a card with a picture of three bowls of ice cream. Each bowl has an invisible button over it, named "chocolate," "vanilla," and "strawberry" (respectively). When the stack terminates, the `HS.Test.Results` file tells you which button was selected. In this case, the "Correct" or "Incorrect" attribute for a button doesn't matter (though it must be one or the other): what you're getting is the information about which item the users chose. If 100 people used this stack, the `HS.Test.Results` file will contain a survey of 100 people—a study of the percentage of which flavor ice cream was preferred.

To use the test results, load the `HS.Test.Results` file into a word processor or disk-based text object. It is also possible to write a SimpleScript program to read the file and organize the data.

Here is a sample of the `HS.Test.Results` file, after running a stack with five questions. Note that the `HS.Test.Results` file is simply appended to each time that a test stack is run, so the file description below represents the *end* of the

HS.Test.Results file. (SimpleScript has a command to delete files which is useful to delete the HS.Test.Results file for a "fresh start.")

name of test stack	Test.Stack
name of student	Bert Savarese
date and time	10/14/71 4:10:42 AM
card ID (not position)	3
name of button with answer	Abraham Lincoln
card ID	4
name of button with answer	Anaheim
card ID	7
name of button with answer	Malice toward none
card ID	2
name of button with answer	Charity for all
card ID	6
name of button with answer	W. Elias Disney
	SCORE
correct answers	3
total answers	4
	-END-

chapter

5

SimpleScript

This section will tell you about SimpleScript, the programming language built into HyperStudio.

Introduction

SimpleScript is a scripting language built into HyperStudio. It allows you to write 'traditional' computer programs (in the SimpleScript language) that can be attached to HyperStudio buttons. When the button is clicked, the program is started.

SimpleScript combines many of the best features of AppleSoft (floating point numerics, common-sense syntax, and ease of use) with the best features of HyperStudio (easy access and manipulation of graphics, text, and sound). With SimpleScript in place, you can make your HyperStudio stacks do more than was ever possible before.

And, yet, SimpleScript is very easy to use. One of the primary concerns when it was developed was to keep its complexity within the realms of the rest of HyperStudio—where 4th graders could find it as easy and fun to use as hardcore hackers.

SimpleScript is most useful when you have a number of operations that you wish to perform, or when you need the powerful capabilities of variables or looping operations. If you want to perform only a very few, basic commands, you may wish to investigate the MiniScript NBA, which allows you to link the functions of several NBAs. More information on MiniScript can be found in appendix D, "Sample NBAs, Extras, and Transitions."

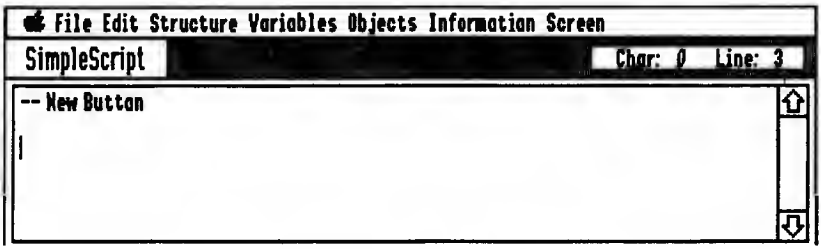
Using SimpleScript

You interact with SimpleScript in the SimpleScript editor. It has pull-down menus containing all of the commands available in the language. Choosing a command from a menu leads you by the hand through the most common syntax for that command. If variables are required, SimpleScript lists the current variables for you. If you need to type in a constant, it will prompt you. This makes it possible to write a complete program without any kind of reference material, even if you've never used the language before!

You can also access SimpleScript through the debugger. The SimpleScript debugger is useful for finding problems in your programs, as well as allowing you quick access to the editor.

The SimpleScript Editor

SimpleScript's built-in editor makes writing a program quick and painless. It provides all of the features of a basic word processor, plus special extras making it easy to develop code.



the SimpleScript editor

You move around your file by using the mouse and the four arrow keys. Additionally, there are several keyboard shortcuts to make using the editor a breeze:

Apple-Right Arrow: move to next word to right
Apple-Left Arrow: move to next word to left
Apple-Up Arrow: move up one window
Apple-Down Arrow: move down one window

Option-Right Arrow: move to end of line
Option-Left Arrow move to beginning of line
Option-Up Arrow: move to beginning of text item
Option-Down Arrow: move to end of text item

From a given location, pressing:

Shift-Option-Up Arrow: selects everything to the start of the document.
Shift-Option-Down Arrow: selects everything to the end of the document.

Control-D: deletes the character to the right of the cursor.
Control-Y: clears all text to the end of the line.

Double-click: selects an entire word.
Triple-click: selects an entire line.

SimpleScript will attempt to reformat your program by indenting lines as appropriate whenever you press the Return or Tab key. The rules that SimpleScript follows for formatting are that every `If` should have matching `Else` and/or `End If` statement, every `Repeat until` statement should have a matching `End Repeat` statement, every `For` should have a matching `Next` statement, and that every `Procedure` should have a matching `Return` statement.

Following these rules, code will be reformatted like this:

Before

```
For I = 1 to 20
Beep
Next
Show message "Annoyed?"

For Punk = 6 to 2
Beep
MohawkMan = 0
Repeat until Plow = 2
Play sound "slither"
Increment Plow
End Repeat
Next
```

After

```
For I = 1 to 20
   Beep
Next
Show message "Annoyed?"

For Punk = 6 to 2
   Beep
   MohawkMan = 0
   Repeat until Plow = 2
      Play sound "slither"
      Increment Plow
   End Repeat
Next
```

You can toggle the automatic indenting feature with the "Auto indent" option, in the Edit menu. The default is on. With auto indenting turned off, SimpleScript will only indent your lines when you press the Tab key or when you exit the editor.

Indented structures, as above, make debugging a little easier—when your program doesn't end in the same column that it started, it's a snap to find out where you forgot a `Next` or `End If` statement.

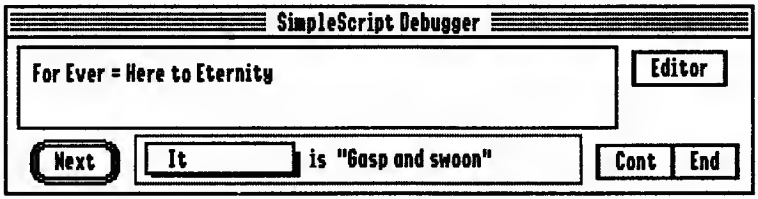
The editor will display the current line offset (from the top of the file) and character offset (from the beginning of the line) in the upper right corner of the screen. Your code may be up to 32k in length, and you can link to another program (associated with another button) at any time—effectively allowing programs of any size.

The SimpleScript Debugger

The SimpleScript debugger is a very powerful tool. If your program has caused HyperStudio to "lock up," you can often recover control by pressing Control-C or

Apple-period. This opens the debugger window, allowing you to watch your program run.

You can also enter the debugger by holding down the Apple key while clicking a button which executes a SimpleScript program.



the SimpleScript debugger

The debugger display will show you the statement about to be executed, and the value of any variable you choose. In addition, it has several controls to help you debug your program. Clicking the "Next" button (or just pressing Return) will allow SimpleScript to execute the line being displayed at the top of the debugger window.

The "Continue" button will allow your program to continue to operate at full speed—it closes the debugger window and continues execution as if you had never triggered the debugger. Clicking the "End" button will make your SimpleScript program stop executing immediately. Finally, the "Editor" button will take you directly into the SimpleScript editor.

The pop-up menu allows you to select a variable to be displayed. The value of this variable is updated in the debugger window as it changes. You may select a different variable to be displayed at any time. Additionally, if you click on the variable's value, a dialog box will open, allowing you to change the contents of that variable "on the fly."

SimpleScript Statistics

Variables

All variables, both character and numeric, are stored as strings. Legal variable names must start with a letter, contain only letters and digits, and may not be longer than 19 characters. They may not begin with a reserved keyword (see appendix B for a list of SimpleScript reserved keywords). Variables may contain any combination of ASCII characters up to a length of 65,535 characters.

Anything that SimpleScript doesn't recognize in a line, it will try to make into a variable. For example, the following statement will try to play the sound file whose name is stored in the variable `BoomBoom`:

```
Play sound BoomBoom
```

Contrast this to this statement, which will try to play a sound file named "BoomBoom":

```
Play sound "BoomBoom"
```

Variables are pre-initialized to a null string (""), which is evaluated as 0. As a short cut, you can assign a variable a numeric value without placing the numeric value in quotes. These two statements are identical:

```
It = "688.0687"
It = 688.0687
```

Math

SimpleScript math is based on Apple's SANE tools, and is accurate to ten digits. SimpleScript supports many different math functions (such as random, round, square, logarithm, and arc tangent).

The language also supports boolean math. Boolean math allows you to evaluate expressions to either a zero or one. With boolean math and a little clever coding, it's possible to save a lot of space in your code. Here's an example program fragment, with and without boolean math:

```
Without boolean math:        It = 80
                             If J = 50
                                  It = It + 40
                             End If

With boolean math:           It = 80 + (40 * (J = 50))
```

In the second version, with boolean math, the computer first evaluates J = 50. If the value of J is equal to 50, the expression is equal to 1, which is in turn multiplied with 40

and added to 80. If the value of J is not 50, the expression is equal to 0, which is in turned multiplied with 40 (resulting in zero) and then added to 80.

Here's another example, this one more complex:

```
Without boolean math:        It = 20
                             If J < 12
                                  It = It + 40
                             End If
                             If K = 90
                                  It = It - 2
                             End If

With boolean math:           It = 20 + (40 * (J < 12)) - (2
                                * (K = 90))
```

Finally, here's an example that centers one of three known strings on the screen. On entry, the string "Flavor" can be equal to "Chocolate," "Vanilla," or "Boysenberry." The appropriate horizontal screen positions to center each of these is 99, 105, and 95, respectively. (The vertical screen location, 103, stays the same regardless of which string is displayed.) Here is the code that does the dirty work:

```
Move to 99 + (6 * (Flavor = "Vanilla")) - (4 * (Flavor =
   "Boysenberry")), 103
Draw text Flavor
```

Boolean expressions may be combined in such a manner that they read more "English-like," by omitting subsequent variables in a group of boolean compares. All of these statements groups are identical:

```
If J > 5 and J < 12...       Repeat until K = 4 or K > 9
If J > 5 and < 12...         Repeat until K = 4 or > 9
```

Non-numbers

Occasionally, a math operation can return a non-number. (For example, the result of dividing a number by zero is a non-number.) These are returned to SimpleScript programs as "Not a Number" strings, in the form "NAN(0*x*)", where *x* is the specific non-number error. You program can check for these illegal operations by simply looking for the letters "NAN" at the beginning of any variable modified by a math operation.

number	*string*	*meaning*
1	"NAN(001)"	invalid square root
2	"NAN(002)"	invalid addition
4	"NAN(004)"	invalid division
8	"NAN(008)"	invalid multiplication
9	"NAN(009)"	invalid remainder or mod
21	"NAN(021)"	attempt to create a NaN with a zero code

33	"NAN(033)"	invalid argument to trig routine
34	"NAN(034)"	invalid argument to inverse trig routine
36	"NAN(036)"	invalid argument to log routine
37	"NAN(037)"	invalid argument to NaN routine

Chunk Expressions

With chunk expressions, you may specify a character, line, or word, or a range of those, for SimpleScript to evaluate. For example, when assigning one variable to another, these are all valid operations:

```
It = Name
It = character 3 of Name
It = words 4 to 6 of InputString
It = line 7 of TextStuff
It = word 12 of gooberPeas
It = lines 1 to 4 of record1
```

When using chunk expressions, a "word" is defined as a group of letters and characters separated by (but not including) spaces, commas, or the carriage return at the end of the line. A "line" is separated by (and does not include) carriage returns. Note that the letter "s" is optional at the end of the statements "characters," "words," and "lines."

The chunk expression programming is intelligent. Retrieving a single word from a string will give you just the word and no surrounding spaces. However, removing that word (with the `Delete text` command) will remove the word and the following space.

Language Reference Definitions

In the language reference below, the following words are used as abbreviations:

expression	a numeric or string expression
variable	variable name
x	an expression that resolves to a numeric value between 0 and 639 (used as a horizontal screen coordinate)
y	an expression that resolves to a numeric value between 0 and 199 (used as a vertical screen coordinate)
statement	any SimpleScript statement
pathname	an expression that resolves to a valid GS/OS pathname. This may be a full or partial pathname.
type	HyperStudio object type—may be "button", "text field", or "graphic"
[]	used to indicate an optional parameter

Apple Menu

The Apple menu contains two things: the first is "About SimpleScript" and tells you a little bit about SimpleScript, including the author, version number, and memory information.

Also located in the Apple menu are your desk accessories, as you have them installed on your system disk.

File Menu

The File menu presents the usual file management options. Many of them are standard for the Apple IIGS desktop interface.

New script ⍁-N

This command erases the current script and prepares for a new script. It also prompts you for a new remark to place at the beginning of the script.

Save and quit editor ⍁-S

With the `Save and quit editor` command, you can use one keystroke to place your SimpleScript program in your HyperStudio stack, and return to HyperStudio. Note that this is *not* the same as pressing ⍁-S from the main HyperStudio program.

Import script from disk ⍁-I

This allows you to load a script (stored as a text or AppleWorks Classic file) from disk. It will place the file into the editor, replacing any script you may have there.

Export script to disk ⍁-E

The sister command to `Import script from disk`, this command allows you to save your current script as a text file on disk. You may want to do this to insert the program in a newsletter or magazine article, or to send a script via modem to a friend.

Page setup

This command allows you to set up your printer preferences via Apple's standard "Page setup" dialog.

Print ⌘-P

The `Print` command allows you to print your script.

Quit editor ⌘-Q

With the `Quit editor` command, you may completely leave the SimpleScript editor *without* saving the current script in your HyperStudio stack. This command will destroy any script that exists in the editor. If you have made changes, and not saved your script, you will be asked if you are sure that you wish to lose the changes.

Edit Menu

This menu contains options that make working with the editor and your script easier and more productive.

Undo ⌘-Z

Currently, this command does nothing. It is provided for certain desk accessories, and may be usable with future versions of HyperStudio.

Cut ⌘-X

The `Cut` command removes the selected portion of your script and places it on the clipboard, where it may be put back into your script with the `Paste` command.

Copy ⌘-C

This command is similar to `Cut`, above, except that it does *not* remove the selected portion of the script—it only places a copy on the clipboard.

Clear delete

With the `Clear` command, you can remove the selected text in your script without disturbing the clipboard. Note that, after using this command, the cleared text is no longer available.

Paste ⌘-V

Selecting the `Paste` command will cause the contents of the clipboard to be placed into your script at the cursor. If a range of text is selected, it will be replaced with the contents of the clipboard.

Debug on ⌘-D

By choosing `Debug on`, SimpleScript inserts the command `Debug on` into your script at the cursor. When the command is executed, the SimpleScript debugger appears.

There is no corresponding `Debug off` statement because, once the debugger is in use, you can turn it off by simply clicking the "Cont" button.

Auto indent

This command toggles the auto indenting of the editor. A check mark will appear to the right of the auto indent option if it is turned on.

Automatic indenting is described in the *SimpleScript Editor* section of this documentation, above.

Find ⌘-F

With the `Find` command, you can quickly search your script for a keyword, variable, or any other sequence of characters.

There are two options: case sensitive, and search from beginning. The case sensitive option controls whether or not the specific case of the characters matters. (For example, with case sensitive marked, a find for "Wow" won't find "WOW" or "wow".)

Normally, the find command searches from the cursor to the end of the script. The search from beginning option allows you to go to the start of your script before beginning the search.

If you want to move to a specific line number, the find command can quickly take you there. Simply search for a pound sign followed by the line you wish to move to. (Searching for "#5" will take you to line five, for example.)

Remark ⌘-R

The `Remark` command changes the line that the cursor is on into a remark by inserting two dashes at the beginning of the line. If the line is already a remark, this command will remove the two dashes. If you have more than one line selected, it will operate on all of them.

Because SimpleScript doesn't have line numbers, remarks can be used in conjunction wit h the `Go to remark` command, to change program flow. Additionally, remarks are useful for placing comments about a piece of complex program code right with the code, so that it can be located quickly. Finally, by placing two dashes as the beginning of *any* line, it will make the line a remark, so that the line will not execute. This is a quick way to stop part of a program from executing without cutting out parts of your program.

Select all ⌘-A

This command simply selects (highlights) your entire script.

Structure Menu

The Structure menu contains all the commands that are the backbone of any language. All commands that influence the flow of control are found here, including looping and branching instructions.

For - Next

Syntax

```
For variable = expression1 to expression2
Next [variable]
```

Examples

```
For It = 1 to 5
    -- draws the digits 1 through 5
    Draw text It
Next

For It = 12 to 3
    -- will beep the speaker nine times
    Beep
Next
```

Description As in AppleSoft, a `For - Next` loop counts from *expression1* to *expression2*, using *variable* as a counter. If *expression1* is greater than *expression2*, the loop will count backwards.

Notes If *expression1* and *expression2* are equal, the loop will execute once. The variable *variable* may be modified during execution of the `For - Next` loop to prematurely end (or extend the length of) the loop.

Sample Use Here's a loop that uses boolean math to wait for the letter "G" from the user:

```
For I = 0 to 1
    -- grab a keypress
    Get event into key1, key2
    -- if key1 isn't G, I is zero
    I = (key1 = "G")
Next
Show message "Thanks, I needed that!"
```

Repeat until - End Repeat

Syntax

```
Repeat until expression
    [statement]
End Repeat
```

Examples

```
Repeat until J = 5
    Increment J
End Repeat
```

```
Repeat until Finished = "Yes"
    Gosub proc "myProcedure"
End Repeat
```

Description The `Repeat until` loop executes program statements until a specific condition is met.

Notes Unlike the `For - Next` loop, above, `Repeat until` gives you the flexibility to set up your own loop counter and increment it by any value (not just one).

Note that `Repeat until` evaluates the expression every time through the loop, *including the first*—it is possible to never execute any of the statements between the `Repeat until` and `End Repeat` statements, if the expression evaluates true immediately.

Sample Use This program beeps when the mouse goes to the bottom of the screen:

```
Y = 0 -- initialize the variable
Repeat until Y > 180
    -- get the mouse loc. in x and y
    Read mouse position into x, y
End Repeat
Beep
```

If - Else - End If

Syntax

```
If expression
   [statement1]
Else
   [statement2]
End If
```

Examples

```
If Joey = "fortuitous"
    Beep
End If

If Ray = "applesauce"
    Gosub proc "fistfight"
Else
    Gosub proc "party time"
End If
```

Description

`If - Else - End If` is one of the most powerful structures in computer languages. It permits you to execute certain statements based on the value of expressions.

If *expression* evaluates to true, all statements between the `If` and the `Else` or `End If` commands will be executed. If *expression* evaluates to false, SimpleScript will scan forward to look for either an `Else` or `End If` statement. If it finds an `Else` statement, it will execute all statements between the `Else` and `End If` statements. Otherwise, no code is executed.

Notes

Note that you do not need to have *statement1* or *statement2*, and the `Else` statement is optional, as well. A combination of `If` and `End If` (with no `Else`) is perfectly legal.

Sample Use

This program accepts commands from the keyboard. Currently, it knows only one command: "sing out".

```
Ask question "What should I do?" with response
If response = "sing out"
    Draw text "On Wisconsin, la, la, la, la, la..."
End If
```

Here's another program, which will play a disk sound called "blat" if you *don't* press Option-O:

```
Get event Key, Modifier
If Key = "O" and Modifier = "Option"
Else
    Play sound "blat"
End If
```

Procedure - Return

Syntax

```
Procedure expression
      [statement]
Return
```

Example

```
Procedure "AnimalSounds"
    -- define the procedure
    Play sound "Cat"
    Play sound "Dog"
    Play sound "ORCA/Whale"
Return
```

Description This is similar to a subroutine in AppleSoft. During program execution, SimpleScript skips all lines between `Procedure` and `Return`, unless the procedure is called with a `Gosub proc` statement.

Notes Internally, SimpleScript looks for procedures starting from the top of your program, so your script will execute faster if you define all of your procedures at the start of your code—however, procedures may be placed anywhere in your program.

Finally, a procedure name can be any length and may contain any printable ASCII character, including spaces.

Be sure to see `Gosub Proc`, below.

Sample Use This short program repeatedly asks the user if they're really sure they want to quit. It uses a procedure to cut down on code size.

```
Procedure "Ask The User"
    Ask choice "Are you sure you want to stop?"
      with response, "Yes", "No"
Return

-- start
Gosub proc "Ask The User"
Show message "I don't think you meant it!"
Gosub proc "Ask The User"
Show message "Now, be honest..."
Gosub proc "Ask The User"
Show message "Okay, if you insist."
```

Gosub Proc

Syntax `Gosub proc` *expression*

Example
```
Gosub proc "bleep"
```

Description With `Gosub proc`, you may temporarily pass program control to another part of your code, which you define with the `Procedure - Return` commands.

Notes See the information on `Procedure - Return`, above.

Go to...

Syntax
```
Go to card expression
Go to button expression
Go to remark expression
```

Examples
```
Go to card "Bert"
Go to button "Main"
Go to remark "top"
```

Description This function provides navigation within the stack (with `Go to card`), as well as simple branching capabilities.

Notes Before going to another card, use the `Set transition` command to specify the screen effect.

Go to button can only find buttons on the current card.

After going to another card, your script automatically stops execution. To get around this limitation, use the `Show card` command.

The `Go to remark` command is useful because SimpleScript doesn't have line numbers. Instead, if you wanted to transfer control to a specific section of program code, you would start that section of code with a remark, then use a `Go to remark` statement to actually branch.

Sample Use This short program asks the user what card they want to go to:

```
Set transition to Rain
Ask question "What card do you want to go to?" with
  response
Go to card response
```

Gosub Button

Syntax `Gosub button` *expression*

Example `Gosub button "bleep"`

Description With `Gosub button`, you may halt execution of the current script, cause another button to be triggered, and then continue with execution of the current script.

Notes Calling a button with `Gosub button` acts exactly as if you had clicked on the button.

Note that this provides SimpleScript with the capability of animation—simply attach your animation to another button, and call it with the `Gosub Button` command!

Currently, you may only call other buttons on the same card.

Pause

Syntax `Pause` *expression ticks/seconds*

Examples

```
Pause 10 seconds
-- wait for 1/2 second
Pause 30 ticks
```

Description The `Pause` function causes your computer to wait for the specified length of time.

Notes Ticks are 1/60th of a second.

Sample Use The following code will wait for as many seconds as the user specifies.

```
Ask choice "How many seconds should I wait?" with
  response, "1", "3", "17", "42"
If response <> "Cancel"
    Pause response seconds
    Beep
    Show message "Done!"
End If
```

End of Script

Syntax `End of Script`

Example `End of Script`

Description The `End of Script` command, when executed, will make your script halt immediately.

Notes Any lines after the `End of Script` statement are ignored.

Sample Use The following code fragment will *not* cause the computer to beep.

```
End of Script
Beep
```

Variables Menu

The Variables menu contains commands to control, manipulate, and modify variables. Both string functions (such as taking strings apart and putting them back together) and numeric functions (like random, sine, and round) are found here.

Global

Syntax `Global` *variable* [,*variable*]

Examples
```
Global Score
Global UserID, Password, PortNumber
```

Description Normally, all variables in a SimpleScript program are "local" to the current script—no other button's script may access them. With the `Global` command, you can specify certain variables to be available to other scripts. By using the `Global` command in *both* scripts where you wish to access a certain variable, it insures that variable will be available.

Note that the `It` variable may not be made global.

Notes You may declare up to eight variables per `Global` command.

Global variables are only cleared when you enter the SimpleScript editor or quit HyperStudio. They will survive the loading of another stack.

Set variable

Syntax *variable* = *expression*

Examples
```
Score = 12
Pleasure = Pain * 2
Voice = "Farm"
```

Description This function (the = symbol) allows you to assign an expression's value to a variable. Note that SimpleScript requires that strings be surrounded by quotes.

Notes Be sure to see the descriptions of the `Combine` and `Get Text` commands.

Sample Use The following code fragment, which uses a boolean math expression, will display the number "1" if you enter the word "shrivel" when prompted.

```
Ask question "What's the word?" with response
-- use boolean math
Me = (response = "shrivel")
-- display result
Show message Me
```

Get length

Syntax `Get length of` *expression* `into` *variable*

Examples

```
Get length of "Hurrah" into Garland
Get length of Score into It
```

Description This function returns the number of ASCII characters stored in a variable.

Sample Use This program will ask you for your name and tell you its length.

```
Ask question "What's your name?" with response
Get length of response into x
Show message "That's " & x & " characters long."
```

Read file
Write file

Syntax

```
Read file pathname into variable1 [with variable2]
Write file pathname from variable1 [with variable2]
```

Examples

```
Read file "HS.Test.Results" into buffer
Write file "report" from buffer
Write file "checksum" from result with errorCode
```

Description

With these two companion commands, SimpleScript has the ability to work with variables in conjunction with disk files. You may optionally add a second variable on the command line, to check for errors.

Notes

One obvious use for this is to display disk-based text in a text item from within a SimpleScript program. You could also use this command to read and interpret the `HS.Test.Results` file. Another function could be to write out a user's input into a disk file for examination in another program.

If you include the optional `with` *variable2* statement at the end of the command, you can find out if a GS/OS error occurred. If *variable2* is zero, no error occurred, otherwise, *variable2* will contain the GS/OS error code (in decimal).

If a full prefix is not specified in *pathname*, the file is saved in the same folder with the stack.

When using `Write file`, SimpleScript checks to see if the file specified already exists. If it does not, SimpleScript creates a new text file. Otherwise, if the file does exist, SimpleScript checks the filetype and auxtype of the old file, then erases the old file and creates a new file using the previous file's filetype and auxtype.

Erase file

Syntax `Erase` *pathname* `[with` *variable*`]`

Examples

```
Erase "HS.Test.Results"
Erase "theFile" with errCode
Erase ignorance with education
```

Description This command allows you to permanently remove a file from the disk. It's most useful for erasing the `HS.Test.Results` file. You may specify an optional variable to return any error code that may have occurred.

Notes You can erase subdirectories *only* if the subdirectory contains no files.

If a full prefix is not specified in *pathname*, the file to be erased is assumed to be in the same directory as the stack.

If you include the optional `with` *variable* statement at the end of the command, you can find out if a GS/OS error occurred. If *variable* is zero, no error occurred, otherwise, *variable* will contain the GS/OS error code (in decimal).

Calculator

Syntax *variable = formula*

Examples

```
It = Random(9)
It = Cos(3)
It = Sine(2) * 360 + Score
```

Description The SimpleScript calculator makes building a formula as easy as operating a desk calculator! As you click buttons, your formula is built on the calculator display. When you leave the calculator by clicking the "Done" button, the formula is inserted into your script.

Notes Use the "variables" pop-up menu to add a variable to the formula, or the "functions" pop-up menu to add a function. The following functions are supported:

`Random (x)`	Returns a random number between zero and *x*. If you specify a value with a decimal point in *x*, you'll get back a random number with a decimal point; otherwise, you'll receive a whole number.
`Sin (x)`	Returns the sine of *x*.
`Cos (x)`	Returns the cosine of *x*.
`Tan (x)`	Returns the tangent of *x*.
`Atan (x)`	Returns the arc tangent of *x*.
`Trunc (x)`	Returns the truncated value of *x*. Truncating a number removes any decimal values. (`Trunc (3.1415)` returns 3; `Trunc (1.9999)` returns `1`.)
`Round (x)`	Returns the rounded value of *x*.
`Root (x)`	Returns the square root of *x*.
`Exp (x)`	Returns the exponent of *x*.
`Sqr (x)`	Returns the square of *x*.
`Ln (x)`	Returns the base-*e* (natural) logarithm of *x*.
`Log2 (x)`	Returns the base-2 logarithm of *x*.

If there's not enough room for your entire formula in the calculator display, you can enter a partial formula and then enter the remainder in the editor. (Of course, you can also enter your entire formula in the editor.)

The calculator will keep track of the number of each kind of parentheses (open and closed) that you use in your formula, and be certain that they are balanced. However, it doesn't (and can't!) check that they are correctly ordered and placed.

Increment
Decrement

Syntax

```
Increment variable
Decrement variable
```

Examples

```
Increment It
Decrement Score
```

Description This function allows you to quickly add or subtract one from a numeric variable. `Increment` is the same as *variable* `=` *variable* `+ 1` and `Decrement` is the same as *variable* `=` *variable* `- 1`.

Get text

Syntax

variable `= characters` *expression1* `[to` *expression2*`] of` *expression3*

Example

```
Score = characters 1 to 3 of It
PorkyPig = word 5 of looneyTunes
```

Description `Get text` allows you to copy specific characters out of an expression into a variable.

Notes This command simply builds an expression using the chunk expressions of SimpleScript, described in the "SimpleScript Specifications" section earlier in this chapter.

Sample Use The following code fragment will display the number "1019".

```
String = "8101966417"
number = characters 2 to 5 of String
Show message number
```

Insert text

Syntax `Insert` *expression1* `into` *variable* `at` *expression2*

Examples

```
Insert "R" into It at character 3
Insert "Eric" into names at line 0
Insert "fairy" into tinkerbell at word offset
```

Description This instruction inserts characters into *variable* before the character position specified by *expression2*, causing *variable*'s length to increase.

Notes Note that *expression2* must be a chunk expression statement.

If the numeric value of *expression2* is zero, the characters are inserted at the beginning of *variable*. If *expression2* is larger than the length of *variable*, *expression1* is inserted at the end of *variable*.

Sample Use This short program prints "Apple II Forever" on the screen.

```
String1 = "Appler"
Insert "e II Forev" into String1 at character 5
Show message String1
```

Delete text

Syntax `Delete characters` *expression1* `[to` *expression2*`] of` *variable*

Examples

```
Delete characters 1 to 4 of Score
Delete characters start to finish of name
Delete character 5+offset of name
```

Description This instruction removes the characters in *variable* from *expression1* to *expression2*, causing *variable*'s length to decrease.

Notes If *expression2* is larger than the length of *variable*, all characters from *expression1* to the end of *variable* are removed.

Because this command uses chunk expressions, you could just as easily re-write it to be `Delete words...` or `Delete lines...`

Sample Use The following code fragment will display the phrase "Preen all!".

```
String = "Support Teen Challenge!"
-- lose "Sup"
Delete characters 1 to 3 of String
-- lose "o"
Delete character 2 of String
-- lose "t T"
Delete characters 3 to 5 of String
-- lose "Ch"
Delete characters 7 to 8 of String
-- lose "enge"
Delete characters 10 to 13 of String
Show message String
```

Replace text

Syntax

```
Replace expression1 with expression2 in variable
```

Examples

```
Replace "Bad" with "Worse" in Status
Replace oldName with newName in nameRecord
```

Description

This instruction searches *variable* for *expression1*. When it finds *expression1*, it removes it from *variable* and inserts *expression2*. This has the effect of replacing one word for another.

Notes

SimpleScript will expand or shorten the length of *variable* as necessary to make *expression1* fit.

You aren't limited to single words. The following examples are all valid:

```
Sheep = "Baa"
Replace "aa" with "aby" in Sheep
-- display the word "Baby"
Show message Sheep

BTog = "dolts"
Replace "t" with "l" in BTog
Show message "Little people like to play with " &
  BTog
```

Find character

Syntax `Find ASCII character` *expression1* `in` *expression2* `with` *variable*

Example
```
Find ASCII character 32 in "Pepperoni Pizza" with
It
```

Description This instruction searches *expression2* for the ASCII character indicated in *expression1* (ASCII is a code which gives every printable character a number). When the character is found, its position is returned in *variable*. If the ASCII character can't be found, *variable* will contain zero.

Notes For your reference, an ASCII chart has been included in appendix B, "SimpleScript Reference."

Sample Use The following code fragment will tell you if you've typed an exclamation mark in a sentence.

```
Ask question "Are you excited?" with response
Find ASCII character 33 in response with It
If It <> 0
    Show message "I guess you are excited!"
Else
    Show message "You don't sound very excited."
End If
```

Combine

Syntax *variable* = *expression* & *expression* [& *expression*]

Example
```
It = "Hey there, " & name
```

Description The `Combine` command allows you to build an expression using the combine operator (an ampersand).

Swap

Syntax `Swap variables` *variable1* `with` *variable2*

Example
```
Swap variables oldScore with newScore
```

Description This instruction swaps the contents of two variables.

Get ASCII

Syntax `Get ASCII of` *expression* `into` *variable*

Examples

```
-- next line returns 82
Get ASCII of "R" into It
-- next line returns ASCII of "1", 49
Get ASCII of 122 into It
```

Description This instruction returns the ASCII value of the first character of the expression you pass it.

Notes This is very similar to AppleSoft's `ASC ()` function.

Make character

Syntax `Make character from ASCII of` *expression* `into` *variable*

Examples

```
Make character from ASCII 103 into It
-- It contains "g"
```

Description This instruction places the character corresponding to ASCII code *expression* into *variable*.

Notes This is very similar to AppleSoft's `CHR$ ()` function. *Expression* may not resolve to a number larger than 127.

Objects Menu

The Objects menu contains commands to manipulate graphic, text, and button objects. Objects include sounds, New Button Actions, and dialog boxes.

Beep

Syntax `Beep`

Examples `Beep`

Description This instruction causes the internal Apple IIGS speaker to beep.

Play sound

Syntax `Play sound` *expression1* `[at` *expression2*`]`

Examples
```
Play sound "Choo.Choo"
Play sound userSound at 14
```

Description This instruction will play a disk-based or standard (embedded) sound file at the default volume level (that is, the volume level that the sound was recorded at). If you specify the optional `at` *expression2*, the sound will play at the volume specified by *expression2*.

Notes SimpleScript will check for a standard sound file with the name you specify in *expression1* first. If that fails, the language will then treat the contents of *expression1* as a pathname and attempt to play that sound from the disk file.

Note that, if a sound was added in advanced user mode, it must have been stored with the standard access method.

expression2 must be between 0 and 15.

Sample Use Here's an extremely tiny jukebox program. It will play a named sound that had been added to your stack or, if no such sound exists (as usually is the case), will play at the disk file specified.

```
-- jukebox
Get filename "What sound do you want to hear?" with
  filename
If filename = "0"
    End of script
End If
Play sound filename
Go to remark "jukebox"
```

NBA

Syntax

NBA *name* [*expression1*, ..., *expression4*] [into *variable1*, ..., *variable4*]

Examples

```
NBA Beep
NBA Floogle into It
NBA Draw shapeID into result
NBA Squeal
NBA Squeal often into telephones
```

Description

The NBA instruction permits access to any New Button Action stored in HyperStudio. You may pass parameters to the NBA, and get information back into the variables you specify.

Notes

The exact syntax of this command (as shown in the examples above) depends on the particular NBA is question. Some NBAs may not require any parameters; other may require a certain number of parameters and may not return any information to your program. Check the documentation that came with the NBA for more information.

Only NBAs that are already used by a button somewhere else in the stack may be used. If only your script will be calling an NBA, you may find it necessary to make a "dummy" button somewhere in the stack to be the "owner" of the NBA.

Show message

Syntax `Show message` *expression* `[with icon` *icon*`]`

Examples

```
Show message "What have I done to deserve this?"
Show message "Please flip the disk over and press
  Return." with icon disk swap
Show message "Whoa, buddy." with icon stop
```

Description This command displays a dialog box with the message specified by *expression*. The dialog box has one button, "OK". You may specify an icon to use, or no icon.

Notes The maximum length of the prompt string, *expression*, is approximately 160 characters.

The pound sign (#), asterisk (*), and caret (^) are not permitted in *expression*. Also, characters created with the option key (such as Option-8, "•") cannot be used.

The values permitted for *icon* are "none" (the same as leaving off `with icon` *icon*), "stop", "note", "caution", "disk", and "disk swap". These are all standard Apple IIGS system icons.

Ask question

Syntax `Ask question` *expression* `with` *variable*

Example

```
Ask question "Who are you?" with It
```

Description This command displays a dialog box containing *expression*, and then *variable* as default text in an edit line. Additionally, the dialog box contains two buttons: "OK" and "Cancel." The user may type a response (up to 32 characters), and then click a button. *Variable* will hold the contents of the edit line when the button is clicked (which may be the same as what it was before the command was executed, if the user didn't change the default text), or "0" if "Cancel" was clicked.

Notes The prompt string, *expression*, may be no larger than 32 characters.

Ask choice

Syntax

`Ask choice` *expression1* `with` *variable*, *expression2* [, *expression3*, ..., *expression12*]

Example

```
Ask choice "Do you like cookies?" with It, "No",
  "Chocolate Chip", "Sugar", "Oatmeal Raisin",
  "Peanut Butter"
Ask choice "What do you do?" with occupation,
  "Dance", "Sing", "Sculpt", other
```

Description

The `Ask choice` command allows you to give a list of options to the user in the form of a dialog box with radio buttons and two regular buttons ("Okay" and "Cancel"). The selection will be returned in *variable*, or, if "Cancel" was clicked, *variable* will contain "0".

Notes

You may specify up to 12 options for the dialog box. The prompt string (*expression1*) may be up to 22 characters; the same limitation is placed on the titles for the radio buttons that you specify (in any other *expressions* specified).

Sample Use

This small program prompts the user for the name of their favorite hypermedia software:

```
-- ask

Show message "What's your favorite hypermedia
  software?"
Ask choice "Select one:" with It, "HyperStudio",
  "HyperCad", "HyperStudio", "PinkWay",
  "HyperStudio", "Turtle-Tick", "HyperStudio",
  "HyperStudio"
If It <> "HyperStudio"
    Show message "You must have made a mistake. Try
      it again."
    Go to remark "ask"
Else
    Show message "Good choice! Yeah!"
End If
```

Ask filename

Syntax `Ask filename` *expression* `with` *variable*

Example `Ask filename "Name your setup file:" with It`

Description The `Ask filename` command displays a standard "put file" dialog, permitting the user to specify a disk location and filename. *Variable* is used as the default filename and *expression* contains the prompt string.

Notes If the user clicks on "Cancel," *variable* will contain "0"; otherwise, *variable* will hold the full pathname to the filename they have indicated.

The prompt string may be no larger than approximately 50 characters, and will be truncated if it does not fit in the dialog box.

Sample Use This small code fragment will prompt the user for a filename, with a default filename of "New.File":

```
filename = "New.File"
Ask filename "Please name the file:" with filename
```

Get filename

Syntax `Get filename` *expression* `with` *variable*

Example `Get filename "Select a data file:" with It`

Description With the `Get filename` command, you may prompt the user for a disk file via the standard "get file" dialog box. *Expression* is used as the prompt string; the full pathname to the file selected is returned in *variable*.

Notes If the user clicks the "Cancel" button, *variable* will contain "0". The prompt string may be no longer than 50 characters, and will be truncated if it does not fit in the dialog box.

Hide object

Syntax

```
Hide type expression1 [on card expression2]
```

Examples

```
Hide button "Selection"
Hide text field FieldID
Hide graphic "plumbing" on card 5
```

Description

This command allows you to cause an object to be "hidden." A hidden object will not appear on the screen again until it is unhidden.

You can unhide an object by using the `Show object` command, or by unhiding it through HyperStudio's object editing tools.

An object will stay hidden even after the stack is saved and reloaded.

Notes

You may pass this command either the object's name or card position in *expression1*, and either the card name or ID in *expression2*. Note that the object's card position value can change depending on the order of objects on the card.

Be sure to see the `Show object` command, below.

Show object

Syntax

```
Show type expression1 [on card expression2]
Show card expression
```

Examples

```
Show graphic "pic3"
Show text field "city" on card "address1"
Show button "betty butter" on card "kitchen"
Show card "deck"
```

Description

The `Show object` command is the companion to `Hide object`, above. It will cause an object previously hidden to become visible again.

Notes

You may pass this command either the object's name or card position in *expression1*, and either the card name or ID in *expression2*. Note that the object's card position value can change depending on the order of objects on the card.

`Show object` has one special option: `Show card` *expression*. This allows you to move to another card and continue execution of the current script.

Delete object

Syntax `Delete` *type expression1* `[on card` *expression2*`]`

Examples
```
Delete button "porkchop"
Delete button buttonhole
```

Description This powerful command permits you to completely destroy text items, graphics, and buttons.

Notes You may pass this command either the object's name or position value in *expression1*, and either the card name or ID in *expression2*. Note that the object's card position can change depending on the order of objects on the card.

Redraw object

Syntax `Redraw` *type expression1* `[on card` *expression2*`]`

Example `Redraw graphic "scoreboard"`

Description The `Redraw object` command will redraw an object on the screen.

Notes You may pass this command either the object's name or card position in *expression1*, and either the card name or ID in *expression2*. Note that the object's card position value can change depending on the order of objects on the card.

Get rect

Syntax

Get rect of *type expression1* [on card *expression2*] into *x1*, *y1*, *x2*, *y2*

Examples

```
Get rect of button "Flaunt" into RLft, RTop,
RRight, RBtm
Get rect of text field myField into a, b, c, d
```

Description

This command returns the enclosing rectangle of the object that you specify.

Notes

You may pass this command either the object's name or card position in *expression1*, and either the card name or ID in *expression2*. Note that the object's card position can change depending on the order of objects on the card.

If the object is not found in the stack, the string "Not found" will be returned in *x1*, *y1*, *x2*, and *y2*.

Sample Use

This small program will beep whenever the cursor is over the text field named "field". You may stop the program by pressing Apple-period.

```
-- next line is an infinite loop
Repeat until 0 = 1
    Get rect of text field "field" into x1, y1, x2,
      y2
-- get mouse coords
    Read mouse into mx, my
    If (mx > x1 and mx < x2) and (my > y1 and my <
      y2)
        Beep
    End If
End Repeat
```

Set rect

Syntax `Set rect of` *type expression1* `[on card` *expression2*`] from` *x1, y1, x2, y2*

Example
```
Set rect of button "vivid" from 5, 5, newRight,
newBottom
```

Description With the `Set rect` command, it's possible to change the enclosing rectangle of an object. This means that you are able to re-size buttons, text items, and graphic objects.

Notes You may pass this command either the object's name or card position in *expression1*, and either the card name or ID in *expression2*. Note that the object's card position can change depending on the order of objects on the card.

Passing a graphic object a rectangle larger than its maximum size will cause it to be stretched to fill the new rectangle.

Get field

Syntax `Get field` *expression1* `[on card` *expression2*`] into` *variable*

Examples
```
Get field "field" into buffer
Get field fieldID into dataLocation
```

Description This command allows you to copy the contents of a text field into a SimpleScript variable.

Notes The easiest way to work with the text, once you've retrieved it into a variable, is with chunk expressions.

You may pass this command either the object's name or card position in *expression1*, and either the card name or ID in *expression2*. Note that the object's card position can change depending on the order of objects on the card.

If the object is not found in the stack, the string "Not found" will be returned in *x1*, *y1*, *x2*, and *y2*.

Set field

Syntax `Set field` *expression1* `[on card` *expression2*`] from` *variable*

Examples
```
Set field It from buffer
Set field "Matt D" from piano
```

Description The sister command to `Get field`, `Set field` allows you to replace the contents of a text item with the contents of a variable. *Expression1* is the text field you wish to set (either the name or ID); *variable* is the variable containing the new contents of the item.

You may pass this command either the object's name or card position in *expression1*, and either the card name or ID in *expression2*. Note that the object's card position can change depending on the order of objects on the card.

Notes A limitation of this command is that it can not place text into group text items.

Sample Use Here's a small program to put the user's name into a text field named "of dreams":

```
Ask question "What's your name, sonny?" with name
Set field "of dreams" from name
```

Sort on text field

Syntax `Sort on text field` *expression* `in` *ascending/descending* `order`

Examples
```
Sort on text field "Pilots" in ascending order
Sort on text field "age" in descending order
```

Description With the `Sort on text field` command, you are able to change the order of cards within a stack. By passing the name (or ID) of a text field in *expression* and specifying the sort direction (*ascending* or *descending*), all cards containing that field will be arranged so that they are in order.

Notes If a card does not contain the text field that you specify in *expression*, it will not be moved. This means that, as an example, you can have a stack laid out with a title card, a help card, and then the cards containing the sort field, and the title card and help card will stay at the beginning of the stack. Additionally, if you have a card in the middle of the stack that does not contain the key text field, it will remain in the middle of the stack and not be moved.

Swap cards

Syntax `Swap card` *expression1* `with` *expression2*

Examples
```
Swap card "old" with "new"
Swap card "red" with "blue"
```

Description This command allows you to change the order of any two cards in the stack: you simply specify them in *expression1* and *expression2* (by either number or name), and they'll trade places.

Information Menu

The Information menu contains commands to release the inner secrets of both your Apple IIGS (such as the time and date) and HyperStudio (such as the name of the current stack or the number of cards in the stack). You may also read the mouse position and wait for an event to occur.

Get stack name

Syntax `Get stack name into` *variable*

Example `Get stack name into It`

Description This command simply returns the name of the stack into *variable*. If the stack has not been saved yet, *variable* will be set to "Untitled".

Get number of...

Syntax
```
Get number of type into variable
Get number of cards into variable
```

Examples
```
Get number of cards into Wild
Get number of buttons into circus
Get number of cards into wonderland
Get number of text fields into chorus
```

Description The `Get number of...` command returns the number of objects that are of the type you specify in *type*. In addition to the normal object types ("buttons", "text fields", and "graphics"), the command can return the number of cards in the stack.

Get card number

Syntax `Get card number into` *variable*

Example `Get card number into location`

Description This command returns the current card's ID in *variable*.

Notes Note that the value returned is not necessarily the ID of the card containing the button which triggered the script.

Get menu mode

Syntax `Get menu mode into` *variable*

Example `Get menu mode into It`

Description With the `Get menu mode` command, you are able to determine the status of the menu bar on the current card. *Variable* will contain either "Visible" or "Hidden".

Notes Note that the value returned is not always the menu status of the card containing the button which triggered the script, but the menu status of the current card.

Get version

Syntax `Get version into` *variable*

Example `Get version into HSver`

Description This command returns the version of HyperStudio.

Notes For HyperStudio 3.0, *variable* will contain "3".

Get score

Syntax `Get score into` *variable*

Example `Get score into It`

Description This command allows you to retrieve the stack's current score (as determined by the testing functions).

Notes For more information on the testing functions, see "Testing Functions" in chapter 4, "Adding a Button."

Get max possible

Syntax `Get max possible into` *variable*

Example `Get max possible into jstryker`

Description With the `Get max possible` command, you can find out the maximum possible score (based on the current stack's testing functions).

Notes For more information on the testing functions, see "Testing Functions" in chapter 4, "Adding a Button."

Get time and date

Syntax `Get time and date into` *variable1, variable2*

Example `Get time and date into Time, Date`

Description The `Get time and date` command returns the system date and time into the variables provided.

Notes The time and date are returned in the format defined by the user in the Control Panel. This is normally "HH:MM:SS *x*M" (where *x* is "A" or "P") in *variable1* and "MM/DD/YY" in *variable2*.

Get border color

Syntax `Get border color into` *variable*

Example `Get border color into Roger`

Description This command returns the current system border color into *variable*.

Notes The border color is returned as a numeric value, from 0 to 15. See appendix B, "SimpleScript Reference," for a list of border color values.

To change the border color, use the `Set border color` command.

Get event

Syntax

```
Get event into variable1, variable2
```

Examples

```
Get event into x, y
Get event into key, modifier
Get event into It, It2
```

Description

The `Get event` command will pause execution of the current script until the mouse is clicked or a key is pressed. It will then return this information into the two variables you specify.

Notes

If the mouse is clicked, the X and Y coordinates of the mouse click will be returned in the two variables. If a key is pressed, the *variable1* will contain the key pressed, and *variable2* will contain one of the following words: "Key", "Apple", "Option", or "Both".

A simple way to determine the event that occurred is to check the value of *variable2*. If it's zero, that means that a key was pressed, since the words "Key", "Apple", "Option", and "Both" don't have a numeric value.

This is also a limitation of the command: you may not click the topmost line of the screen (line 0). A way to work around that limitation is to check for a specific keypress or a mouseclick, as the sample program does below.

Note that the nature of this command (returning a word in a variable) makes it case-sensitive, in the respect that the words "Key", "Apple", "Option", and "Both" must be spelled exactly like that and have the first letter capitalized.

If the user has key translation turned on in the Keyboard CDEV, Option key combinations will return special characters rather than regular characters with the option key pressed. (For example, if you were to run the program below with key translation turned on, pressing Option-8 will not return "8" in `key`, but instead "•". In both cases, "Option" will be returned in `modifier`.)

Sample Use

This short snippet will beep when Apple-B is pressed, or present a dialog box if any other event occurs (such as a mouse click or other keypress).

```
Get event into key, modifier
If (key = "B" or key = "b") and modifier = "Apple"
    Beep
Else
    Show message "You didn't press Apple-B!"
End If
```

Screen Menu

The Screen menu contains commands to position and draw text or shapes directly on the card. It also contains commands that set up transitions for moving from screen to screen, and change the Apple IIGS border color.

Move to

Syntax `Move to x, y`

Examples
```
Move to 300, 30
Move to horiz, vert
```

Description This command allows you to specify the position for a subsequent `Draw text` command.

Read mouse

Syntax `Read mouse position into x, y`

Example `Read mouse position into Pam, Addie`

Description The `Read mouse` command returns the current horizontal and vertical position of the mouse, regardless of whether or not the mouse has been clicked.

Notes Unlike `Get event`, this command does *not* wait for an event to happen. It returns *x* and *y* immediately.

Sample Use This demonstration program waits until the mouse cursor is over a graphic object named "hat," then causes the hidden object named "rabbit" to appear.

```
-- be sure it's not there
Hide graphic "rabbit"
-- be sure to use a funny voice when reading next
  line out loud
Draw text "Hey, Rocky, watch me pull a rabbit out
  of my hat!"
Get rect of graphic "hat" into x1, y1, x2, y2
Repeat until (mx > x1 and mx < x2) and (my > y1 and
  my < y2)
    Read mouse position into mx, my
End repeat
Show graphic "rabbit"
```

Set text color

Syntax `Set text color to` *expression*

Example
```
Set text color to 15
```

Description With the `Set text color` command, you may specify the color for subsequent `Draw text` commands.

Notes A list of text colors can be found in appendix B, "SimpleScript Reference." When building this command by selecting it from the pull-down menu, the *<which?>* option will present a color palette, allowing you to simply click the color you wish to use.

SimpleScript doesn't just use "red," "blue" or other color names because the actual color may vary if you are using custom colors on a card. However, if you like, you may set your own variables. For example,

```
Red = 15
Set text color to red
```

Set font

Syntax `Set font to` *fontname size* [*style*]

Examples
```
Set font to Times 24
Set font to Venice 14 bold, italic
```

Description This command allows you to specify a font for the `Draw text` command. You may specify a font name, size, and optionally, a style.

Notes Valid font styles are "bold", "underline", "italic", "shadow", and "outline". If you'd like plain text, simply leave off *style*. Additionally, you may specify more than one different style by separating them with commas.

Choosing this command from the pop-up menu allows you to select the font using the standard Apple IIGS font selection dialog.

Draw text

Syntax

```
Draw text expression [;] [:]
```

Examples

```
Draw text "Hello"
Draw text name;
Draw text "Hello, " & It:
```

Description

The `Draw text` command gives you the ability to paint text on the super hi-res screen, using the text color and font that you've already specified.

Notes

This command mimics AppleSoft's print statement by advancing the internal cursor location (where text will be drawn again) below the text you paint. (For example, executing the command `Draw text "Hello"` twice will place the word "Hello" on the screen twice, one immediately below the other.)

By placing a semicolon at the end of *expression*, `Draw text`'s behavior can be modified so that the internal cursor position is not changed—it will remain at the end of the string drawn. If you place a colon at the end of *expression*, the internal cursor position will be reset to the same location that it was before the command was executed. With this, it's easy to immediately erase the text by simply re-drawing it in the background color.

Sample Use

Here's a small fragment to flash the word "Blastoff!" several times:

```
For It = 1 to 5
    Set text color to 4 -- red
    Draw text "Blastoff!":
    Beep
    Set text color to 15 -- white
    Draw text "Blastoff!":
Next It
```

Set border color

Syntax `Set border color to` *expression*

Example `Set border color to 5`

Description This command allows you to change the color of the screen border. Permissible values for *expression* are 0 through 15.

Notes Try to leave the system as you find it—find the old border color and save that value before changing it to a new color. When you're done with the new border color, change it back to the original color.

A list of border colors can be found in appendix B, "SimpleScript Reference." Selecting the command from the pull-down menu will allow you to select the color by simply clicking it.

Refresh screen

Syntax `Refresh screen [with` *expression*`]`

Example `Refresh screen`

Description This command causes HyperStudio to re-draw the current card, and all objects on the card (except those that are hidden).

Notes If you have drawn text or graphics on the screen, they will be erased by this command. To keep them permanent, you need to move off of the card (that is, change the current card.)

If you specify *expression*, it will be used as the new background color for the screen. (*expression* may be from 0 to 15.)

Show cursor

Syntax `Show cursor`

Example `Show cursor`

Description With the `Show cursor` command, you can make the cursor become visible.

Notes This command is most useful after a `Hide cursor` command, described below.

Hide cursor

Syntax `Hide cursor`

Example `Hide cursor`

Description This command makes the cursor disappear.

Notes You can make the cursor re-appear with a `Show cursor` command, described above.

Set next transition

Syntax `Set next transition to` *transition*

Example `Set next transition to Dissolve`

Description This command allows you to change the screen transition used by the `Go to card` and `Show card` commands.

Notes Selecting this command from the menu will give you a list of all available transitions. A list of transitions built into HyperStudio can be found in appendix B, "SimpleScript Reference."

Set line size

Syntax `Set line size to` *expression*

Example `Set line size to 4`

Description With the `Set line size` command, you may specify the thickness of lines (in pixels) used with SimpleScript's draw commands.

Notes *expression* may be no larger than 24.

Set pen mode

Syntax `Pen mode is` *Copy/And/BIC/XOR*

Example `Pen mode is Copy`

Description The `Set pen mode` command allows you to specify the way that the drawing commands affect the background. (Note that the word following `Pen mode is` is not an expression or constant but is simply one of the four pen modes: `Copy`, `And`, `BIC`, or `XOR`.)

Notes The four pen modes can be summarized by the following tables:

Copy		Pen off	Pen on
Destination	*off*	off	on
	on	off	on

BIC		Pen off	Pen on
Destination	*off*	off	off
	on	on	off

And		Pen off	Pen on
Destination	*off*	off	off
	on	off	on

XOR		Pen off	Pen on
Destination	*off*	off	on
	on	on	off

What this table means is that, when SimpleScript draws items, it compares each pixel against the screen pixel where it is to be drawn. In the *Copy* mode, if the pen (original image) is "on" (that is, if the original pixel is not the background color), that pixel will be copied to anyplace on the destination, regardless of the status of the destination pixel (on or off). The effect is that the image is simply drawn on top of anything on the screen.

However, in the *And* mode, the source image is only drawn wherever the pixels on the screen (destination) are turned on. The result of this is that the image you are drawing only appears wherever something painted on the screen already exists.

By using the tables above, you can determine how the screen will look after you issue your drawing command. Another easy way is to simply experiment with a short program that draws one image on top of another, in each of the four different modes.

Fill flag

Syntax `Fill flag is` *on/off*

Example `Fill flag is off`

Description This command allows you to change HyperStudio's internal fill flag. The fill flag affects how boxes and ovals are drawn with SimpleScript—with the fill flag on, they are filled with the current color; otherwise, they are only outlines.

The word following `Fill flag is` is not an expression or constant—it's just the word `on` or `off`.)

Set line color

Syntax `Set line color to` *expression*

Example `Set line color to 31`

Description With the `Set line color` command, you may specify the color SimpleScript uses to draw dots, boxes, ovals, and lines.

Notes Values from 0 to 15 select a solid color; values from 16 to 31 select patterns. Selecting this command from the pull-down menu will allow you to select the color by simply clicking it.

A list of colors can be found in appendix B, "SimpleScript Reference."

Draw dot

Syntax `Draw dot at` *x, y*

Example `Draw dot at 120, 125`

Description This command draws a dot on the screen at *x, y* using the line color, line thickness, and pen mode already specified.

Notes The dot may or may not be one pixel in size, depending on the line thickness setting.

The `Refresh screen` command will erase anything drawn on the screen with this command.

Draw line

Syntax `Draw line from x1, y1 to x2, y2`

Example `Draw line from 40, 50 to 600, 170`

Description The `Draw line` command allows you to draw a line on the screen from *x1*, *y1* to *x2*, *y2* using the line color, fill flag, line thickness, and pen mode already specified.

Notes The `Refresh screen` command will erase anything drawn on the screen with this command.

Draw box

Syntax `Draw box from x1, y1 to x2, y2`

Example `Draw box from 30, 30 to 600, 170`

Description With the `Draw box` command, you can draw a box on the screen from *x1*, *y1* to *x2*, *y2* using the line color, line thickness, fill flag, and pen mode already specified.

Notes The `Refresh screen` command will erase anything drawn on the screen with this command.

Draw oval

Syntax `Draw oval from x1, y1 to x2, y2`

Example `Draw oval from 400,100 to 500,120`

Description This command draws an oval within the coordinates that you specify, using the line color, fill flag, line thickness, and pen mode already specified.

Notes The `Refresh screen` command will erase anything drawn on the screen with this command.

chapter

6

Sound Shop™

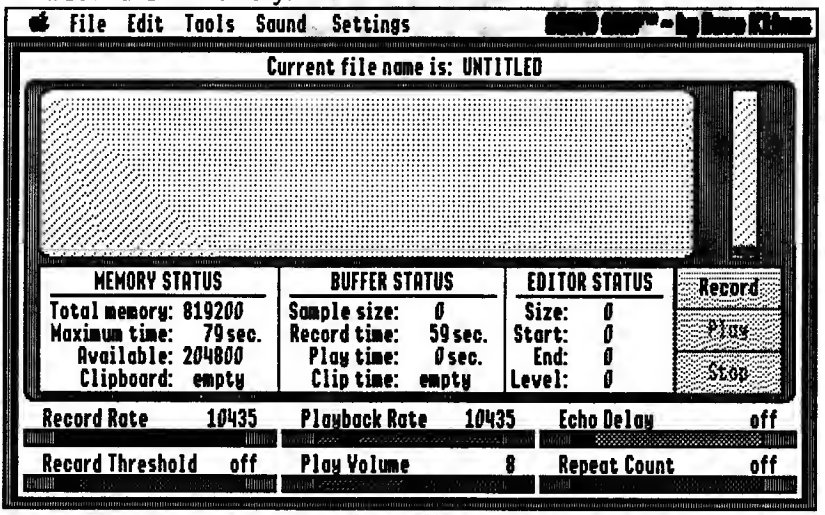
Sound Shop is the accessory program included with HyperStudio which allows you to work directly with your sound files. You may apply special effects, examine waveforms, and apply filtering algorithms to a sound.

Sound Shop

HyperStudio lets you record a sound while creating a stack. The Sound Shop lets you digitize, edit, and save sound files for use with HyperStudio. It can also convert sound files from other sources, such as public domain sound samples, and sounds recorded with the Applied Engineering Sonic Blaster, Audio Animator, or other digitizing systems.

To run Sound Shop, click the Sound Shop icon on the HyperStudio Home Stack. Sound Shop can also be launched from the Finder, ProSel, or other program selector.

After starting Sound Shop, he first thing to notice is the screen layout. The window in the top half of the screen is where the sound waveform is displayed. The window is blank until a sound is in memory.



Sound Shop

Below the waveform window are three status boxes. They show important information about how much memory is available for recorded sounds, how much time that represents at the current record and playback rates, and other useful values.

The scroll bars at the bottom of the screen controls the sound in terms of volume, record/playback rates, "echo" effect, repeat count, and recording threshold value. (All of these are explained more fully, a little later in this section.)

To the right of the status boxes are three buttons: "Record" starts the recording process, "Play" starts the playback, and "Stop" stops the playback. Above these buttons is an "input level" indicator. It checks the current level of the sound input device (which might be the HyperStudio microphone, or a tape player). The "Play" and "Stop" buttons are disabled until a sound is in memory.

A Brief Test of the System

You should have already installed the HyperStudio sound recording card (digitizer) and microphone. If not, please install them now—installation instructions are at the beginning of this manual.

The microphone should be plugged into the back of the computer. When it is working properly, you will see a green vertical bar just above the "Record" button. This is the level indicator, and it tells you if a sound input device is connected. The bar will be red if the microphone is not attached, or if the two-wire plug inside the computer was not properly attached to the "fan" connector.

The green bar should move up and down as you speak into the microphone. (This helps you set the right level *before* you start recording. The bar does not move while recording.) The bar turns red whenever the sound is too strong for a good recording. (For example, shouting into the microphone or turning the volume too high on a tape player.) Under normal conditions, the bar should rarely change to red.

Check the Record Rate and Playback Rate controls at the bottom of the screen. For now, both of these values should be 10435. The rate value tells you how much memory is required for each second of sound recorded. Here, you will use 10.4K of RAM memory per second. This is a good rate for recording voice. There will be more on recording rates later in this chapter; for now, the important thing is to make sure the values are the same for both record and playback.

The other settings should be at their default values:

Echo Delay	off
Record Threshold	off
Play Volume	10*
Repeat Count	off

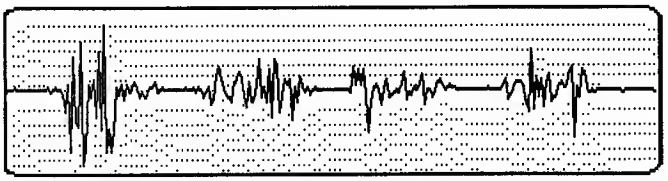
* The playback volume has been preset to 10 for best results with the Apple IIGS internal speaker. If you are using the optional HyperStudio powered speaker, adjust the playback volume to 10.

To record your voice: click the "Record" button with the mouse. When you see the screen border turn red, count, "One, two, three, four," into the microphone. Pause a little between each number. (Note that the level indicator is not active during a recording.)

When recording with the microphone, you must speak firmly for a good recording. A loud or shouting voice is not required. Also, be sure to hold the microphone very close to your mouth. The right level is sometimes called a "radio voice." That's the way a radio announcer speaks into the microphone for clarity and volume.

Press Return to stop recording. (The "Stop" button is for stopping the playback of a sound, not for stopping the recording of a sound.)

After a brief pause, the sound's waveform will be drawn, and will look similar to this:



The data you have just collected is a *sound sample,* and the waveform represents a digital recording of your voice. At the current rate setting (10435), the computer is checking the microphone input a little more than 10,000 times each second, and records a value each time it checks the microphone. The graph represents the sound sample, and each group in the waveform is one of the words you spoke.

(After much research, we have determined that the ideal record rate in Sound Shop is 26320. This value is in tune with the frequencies that your computer operates and as a result, will give you extremely clean sound recordings. If you're doing some recording and need very high quality, this value is recommended. You can also get very good quality by using exactly half this value, 13160.)

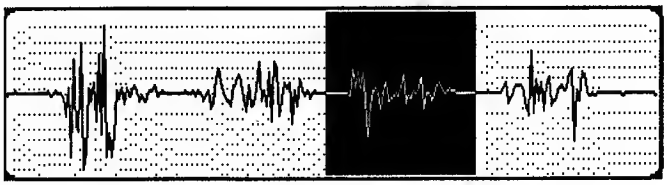
To hear your recording, click the "Play" button. Play your sound several times. Try pressing the "Stop" button in the middle of the playback.

Now let's see what Sound Shop will let you do with your recording.

Playback Rate: Use the mouse to change the Playback Rate (the scroll bar control near the bottom of the screen). Try values of about 20,000 and 5,000. If the Playback Rate is faster than the Record Rate (indicated by the Playback Rate value being *larger* than the Record Rate value), your sound's pitch will be higher and the sound will play faster, producing a chipmunk effect.

If the Playback Rate is slower, the pitch will be lower and the sound slowed down. When you save the sound sample, the Playback Rate setting will be saved with the file.

To play part of a sample, return the Playback Rate value to 10435. Drag the mouse to highlight the entire third wave group. The waveform display should look something like the picture below.



This is how you *select* part of the waveform. Now press the "Play" button. Notice that only the word "three" is played. Whenever you select a portion of a sound, Sound Shop plays only the selection.

(Once you have a range in your waveform selected, you can "grab" either edge of the selected area, and drag it to the left or right to adjust just the beginning or end of the range without affecting the other side. If you put the mouse cursor in the middle of a selected range, you can drag the "window" of that selection to the left or right, as well.)

Volume: This controls the *playback* volume of your sound sample. It does not affect the recording volume. The Volume setting will be saved when you save your sample to

disk, so each sound will have a default volume setting when you first bring it into a stack. Try changing the volume now to see how it affects the sound. Very high or low volume settings will not sound as good as moderate settings, due to the way sound is reproduced on your computer.

Repeat Count: This controls how many times the sound will be repeated when it plays. This might be used with sound effects like a gunshot or footstep. You can get interesting effects without taking up the amount of memory that repeating the same sound effect several times would require. Try changing the Repeat Count to 3, and click "Play." You should hear the word "three" said three times.

Occasionally, due to a bug in Apple's sound tools, sounds of certain lengths and playback rates will be caught in an endless loop. If you play another sound, however, or click "Stop," the sound will stop playing. This problem should be fixed in a future version of the Apple IIGS System Software and can be corrected by lengthening or shortening the sound sample.

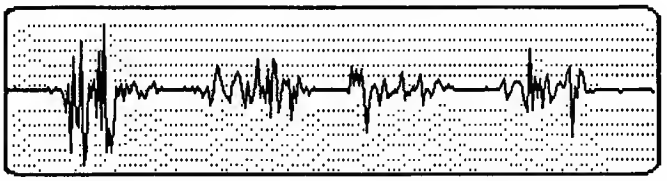
Echo Delay: This controls a delay within the computer between the repeated playing of the sample. Each repeated action is started while the previous playback is still playing. The resulting sound depends on a combination of the sample length, the Playback Rate, and the Echo Delay value. Even slight variations in the Echo Delay value can produce different effects. In general, the echo effect will vary between a "close" echo (like singing in the shower) and a "larger" echo (like that in a stadium).

The best approach is to start with a value of about 40. Play the sound sample. Adjust the Echo Delay to 30 or 50, and try again. Experiment until you get the effect you want.

Remember to set the Echo Delay back to "off" before trying the features described next. It doesn't hurt anything to combine it with other functions, but there will be less confusion if you leave it off for now.

Record Threshold: Ordinarily, recording starts soon after you click the "Record" button. However, you might want to start recording when the sound actually starts. The Record Threshold lets you specify a number value. It represents how loud the sound should be to trigger the recording process.

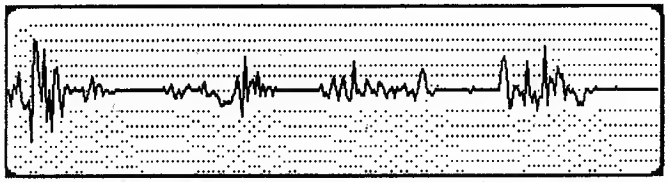
To see how it works, first do a recording with the Record Threshold set to "off." This means recording starts as soon as you click the "Record" button. (If you still have "three" selected, click someplace else in the waveform to deselect it.) Click the "Record" button, wait for a few seconds, and then say "one, two, three." Press the Return key to stop the recording, and the waveform display should look like this:



The flat area at the beginning of the waveform is the time you paused before starting to count. This delay wastes memory. If the sound you are trying to record is not predictable, you could use all the available memory just recording the waiting period.

Now set the Record Threshold to 10. Click "Record" again. This time, the screen border will turn green. Wait a few seconds, then start counting again. As soon as you start

speaking, the screen border should change to red—this shows that the actual recording process has begun. Press Return to stop recording, and the waveform display should look like this:



No space is wasted at the beginning of the recording. The Record Threshold value you choose depends on the sound you are recording. The center line of the waveform window has a value of zero, and the top and bottom of the window represent values of +127 and -127, respectively. If the sound you are recording is relatively loud, you can use a high threshold value. If you are trying to capture a quieter sound, a smaller value like 5 or 10 might be better.

HyperStudio uses a threshold value of 32 when recording sounds.

Now let's look at the menu functions in Sound Shop.

File Menu

This menu is used to save sounds you've recorded with Sound Shop or HyperStudio (that are disk-based), and to load the sounds back in Sound Shop or HyperStudio.

New (⌘-N): This erases whatever sound is currently in memory.

Note that, if you have recorded a new sound or made a change to an existing file, then you will be prompted to save your changes before the file is erased.

Open (⌘-O): This loads a sound file from disk. When selected, you are presented with the standard file dialog. Try loading one of the sample sound files on the `/HS.Sounds` disk.

Note that, if you have recorded a new sound or made a change to an existing file, then you will be prompted to save your changes before loading a new sound.

Save (⌘-S): This is used to quickly save a file under the name that it currently has. If the file isn't named yet, selecting "Save" will be the same as selecting "Save As..." (described below) and will give the file a name. Once the file name is set, this menu item will become "Save *filename*."

Save As...: This is used to save a new file or an existing file under a new name. As with the "Open" command, you will be shown the current disk directory, and asked for the name that will be used when saving the file. "Save As..." can be used to assign a new name to a recording you have changed without overwriting the original.

It is *very* important to save your work frequently. Because of the amount of memory that would be required to set aside a copy of editing actions performed on sounds, an "Undo" option is usually not practical. To get around this, save your work before you make any changes to it, such as deleting a segment or applying a special effect. If the results of the changes you make are not exactly what you want, you can use "Revert" (below) to go back to your most recently saved version.

You have a choice of four different sound formats to save your file:

Standard Format:	this is the normal HyperStudio "Sound Shop" sound file format.
Audio IFF (AIFF):	this format is for exchanging sound files with other computers (AIFF is a "generic" sound file format, much the way that GIF is a "generic" graphic file format). In general, unless you want to use a sound file on another computer, you won't need to save your files as AIFF.
Compressed music:	this option compressed data at the ratio of 8:3 (that is, the data is compressed to 62% of its original size). It is the suggested rate if you want to compress music.

Compressed voice: this option compressed data at the ratio of 8:4 (that is, the data is compressed to 50% of its original size). The resulting file will sound a little rougher than something compressed with the "Compressed music" setting, above.

HyperStudio can load all of the file formats above except AIFF.

Revert: This command simply reloads the current sound file from the disk. It is useful if you make a change to a sound file and then decide that you didn't like the change.

Quit (⌘-Q): Choose this when you want to quit Sound Shop. If you have not saved your work, you will have the chance to save the file before you quit.

Edit Menu

This menu contains commands for editing your sound sample.

Undo (Ó-Z): This features is not used by Sound Shop, and is here for any desk accessories you have which may use it.

Cut, Copy, and **Paste (Ó-X, Ó-C,** and **Ó-V):** These three functions are used to remove or copy a portion of a sound from one position in the sample to another. You can even move parts of a sound between different files. The sound remains on the *clipboard* until you quit Sound Shop, or until "Cut" or "Copy" is chosen again.

There are also special effects that can be done by combining these editing functions with some of the functions in the Tools menu, described a little later.

"Cut" removes the highlighted part of the sound sample, and places it on the clipboard. To use it, select part of a sound sample and choose "Cut" from the Edit menu. After a brief pause, the screen will be redrawn with the previously highlighted section cut from the waveform and placed on the clipboard. To hear what is on the clipboard, choose "Play Clipboard," also in the Edit menu.

The "Copy" function is exactly like "Cut," *except* it does not remove the sound from the waveform—it simply moves a *copy* of the selected sound to the clipboard.

To use "Paste," first click someplace in your sound to set a marker bar, indicating where you wish the contents of the clipboard to be placed. Then, select "Paste." There will be a brief pause while the sound on the clipboard is pasted into the sound sample at the marker bar, and the waveform will be redrawn.

The copy function puts a copy of the selected portion on the clipboard. It does *not remove* it from the original sample. You can use the copy command to get part of the sound from one file, load a second file, and paste the sound into the second file.

Note that if you are copying and pasting between two sound files, and the playback rates in each file are different, you will get accordingly strange results!

There is really no way to get around this problem. Try to minimize the number of different recording rates you use. Sound Shop provides a variety of record and playback rates for compatibility with sound files created with other hardware products. The 10435 recording rate will do fine for most voice recordings. For music or sounds with many high frequencies, a rate of 16398, or perhaps 22824, will work well. If you limit yourself to using just these rates, you'll have much more flexibility in combining parts of one sample with another.

HyperStudio uses 8070 as the "voice" setting, and 16398 as the "music" setting.

Clear: The clear function is a lot like cut except the clipboard is not used. For example, suppose you had the word "three" on the clipboard. You want to delete the word "two" from the middle of the sample without losing the word "three." Select the wave for "two" and choose "Clear." The selected portion will be removed from the waveform. "Three" will still be on the clipboard. The length of the entire sound is shorter.

Play Clipboard: This plays whatever is on the clipboard from the last cut or copy operation. Use it to verify what is on the clipboard before pasting it somewhere else.

Clear Clipboard: This erases whatever is currently on the clipboard. The main use for this would be in those cases where you wanted to use as much memory as possible for an operation (such as a long recording). This is most likely to be of use only when the "Maximum Memory" option is in effect. (For more information on the "Maximum Memory" option, see the description below.)

Tools

The Tools menu provides additional editing functions. They can produce interesting effects and help you create the best sample possible.

If a specific part of the waveform is *not* selected, the action will be performed on the entire sample. That is, *unless* you select part of the waveform first, the action you specify will be applied to the entire sample.

If you select just part of your sound sample, only that part will be affected.

Erase (Ó-E): This clears the selected part of the waveform back to zero (the center line). It does not shorten the length of the entire sound. This is used most often to erase just part of a sample and edit out a flaw. To see how this works, we'll record something with a deliberate "pop," then we'll use the Erase command to remove it.

Record "One, two, three, four" again. This time, tap the microphone once with your fingertip between "two" and "three."

Now, play the recorded sound. You should hear a "pop" between the words "two" and "three." This shows in the waveform as a sharp spike in the middle.

Use the mouse to select just the spike, click the "Play" button to verify that you've selected only the "pop."

Choose "Erase" from the Tools menu (or press Apple-E). The selected part of the waveform will be erased back to the center line. The "pop" has been removed.

The Erase function is handy:

- When recording sounds from noisy sources, or
- When you want to remove a portion of a sound, but don't want to change the overall length of the sample by using the "Cut" or "Clear" commands.

Fade Up and **Fade Down**: **(Ó-U** and **Ó-D):** Use these commands to taper the volume of a sound at the beginning or end. "Fade Up" starts the volume at zero, and then "ramps up" to the normal loudness over the range you have selected. This could be used to edit a sample that has been "sliced" from the middle of a longer sound—by doing a "Fade Up" at the beginning, your sample will sound more natural.

"Fade Down" is just the opposite: it is used to fade out the sound over the range you have selected. These effects can be used anywhere in the waveform, not just at the beginning and end. Note that either fade should be used over a range of at least one second, for the most satisfactory results.

Filter (Ó-F): This is a subtle "smoothing" function. It can take some background hiss out of a noisy sample. To use it, select the portion of the sound sample you want to clean up. (Like the other functions in the Tools menu, if nothing is selected, the entire sample is filtered.) Choose "Filter" from the menu, then click "Play." You may have to filter it several times to produce a noticeable effect; however, filtering too many times will make your sample sound dull or muffled.

Layer (Ó-L): This takes whatever is on the clipboard and *adds* it to the existing sound at a selected point. With it, you can produce your own echo effect. You can also mix different sounds.

For example, you might want to combine two sounds: one of falling rain and another of thunder. To do this, load your thunder sound file, and use the "Copy" function to copy a section you like to the clipboard.

Next, open your rain sound file. You want to add the thunder "in the background." Click the mouse in the middle of the rain sample to place the marker bar, and then choose "Layer." The sound on the clipboard will be combined with the rain sample at the marker point. The two waveforms will be added together, producing a mixed sound.

You can also use "Layer" to produce an echo effect. Make a fresh recording of "One, two, three, four." Select the word "three" from the waveform.

Now look at the Editor Status box, just to the left of the "Record" button. Note where your selection starts (the "Start" value). Choose "Copy" from the Edit menu to copy the word "three" to the clipboard.

Click once with the mouse a little to the right of where you selected "three" a moment ago. For example, if your first selection started at 21000, try putting the marker bar at 22000. Finally, choose "Layer."

Click "Play," and the result should be a kind of "hollow" sounding echo. Putting the marker bar further into the word "three," and choosing "Layer" would have produced a "larger" echo sound.

Layering works better with higher record/playback rates. At lower rates, you may have problems with the combined sound being quieter than the original files. Higher rates avoid this problem, but at the cost of more memory.

Note that, if you are layering two sound files together and the playback rates in each sound are different, you will get accordingly strange results!

Redraw (Ó-W): This will redraw a selected portion of the waveform, or restore the original view after viewing just a portion.

To see how this works, record yourself saying "one, two, three, four," and then select just the word "one." Select "Redraw" from the Tools menu (or type Apple-W). You'll notice that the waveform display has changed to blue, and that you've "zoomed in" on just the word "one." If you click "Play," you'll hear just the word "one."

Now select just the first half of the waveform for "one." (Highlighting will be yellow instead of the normal blue.) Choose "Redraw" again, and then "Play."

You should have just the "wa" sound in "one." With "Redraw," you can select and reselect portions of the waveform until the window shows each individual "bump" that creates your sound. A very small sample may only be a few hundredths of a second. The smallest sample that Redraw will show is 256 data points from one side of the window to the other.

To restore the original waveform display, choose "Redraw" *without* selecting a part of the waveform.

Reverse (Ó-B): This function reverses the selected portion of the waveform, or the entire sample if nothing is selected. It is mostly of novelty value. However, if your favorite local radio station is having a contest to recognize popular songs played backward, it may represent some real income potential to you!

One impressive effect, a "pre-echo," can be achieved by reversing a selection, producing an echo effect on the reversed sound with the "Layer" command (explained above), and then reversing the sound a second time to make it play forward. The sound in your sample will begin to echo *before* the main part of the sound is played.

Stutter (Ó-T): This command is similar to selecting a range of a waveform, using "Copy" from the Edit menu, and then "Paste" several times. Try selecting just the beginning of a word in a waveform and then using "Stutter" to produce a Max Headroom-like effect.

Sound Menu

This menu gives you the equivalent of the "Play" and "Record" buttons. It also offers an alternative Record mode.

Stop and Play (⌘-H and ⌘-P): These are the equivalents of clicking the mouse on the screen buttons. The keyboard equivalents may be more convenient to use when doing repeated operations.

It is important to mention again that the "Play" button *only plays the selected part of the waveform.* This is useful when you are about to copy or paste—you can make sure you have selected just the portion you want. Also, sometimes you want to concentrate on just a portion of a recording. This makes it possible without having to listen to the whole waveform each time you select "Play."

Record Modes (⌘-R and ⌘-M): Recording can be one of two methods. In the usual, or *default,* mode (Record Mode 1), the recording starts as soon as you press the "Record" button. The recording stops when you press Return or when all available memory is full.

This works fine for short sounds. For example, if you are recording a sound from a tape, there is a space at the beginning where you start the player, and a space at the end where you let the tape play for a bit before stopping the recording. Both "dead areas" in your sample are easily edited by using the "Cut" or "Clear" functions.

But what about those cases where there are many continuous sounds and you want to get just the right part of the sample? Perhaps you want part of a cat fight outside your window. When you start the recording process, it will stop on its own in a short time. You could miss the perfect yowl just as you were restarting the recording for the fourth time.

Record Mode 2 is the solution to this problem. Record Mode 2 is a *continuous* recording operation. It doesn't stop until you press a key. When you stop the recording, the *last* sounds you heard will be at the end of your sample. The Buffer Status window will show how many seconds are before that. You can cut or clear most of the sound at the beginning to get just the part you want to save.

To tell Sound Shop you want to use Record Mode 2, select "Record Mode" in the Sound menu. You'll get a dialog box to choose between the two recording modes. (The current recording mode is indicated by a checkbox in the Sound menu.) The selected mode will remain in effect until you quit Sound Shop.

Selected Record: Although not a specific menu item, this function can come in very handy. If you select just a portion of your sample, and then choose "Record," Sound Shop will replace *just the selected part* with the new recorded sample. You can use this to "patch" a sound sample if you wish. Note that this simply replaces the selected portion. It does not lengthen or shorten the selected area. If you patch-record a shorter sample than the selected area, any area not recorded over will retain whatever sound was there. Similarly, if you try to record for a period longer than the selected area, the record mode will automatically stop, and you will not record anything after that point.

Important: Recording a new sound will always clear the clipboard. This is so Sound Shop is able to offer you the most memory for your recordings.

Settings Menu

This menu allows you to set and reset various Sound Shop parameters. The options are:

Current File: Use this to revert the sample size and playback rates to their original (i.e., most recently saved) values.

Startup Defaults: Choosing this sets the sample length to approximately 2/3 of the total free memory in the system. It restores the record and playback rates to the startup values.

Maximum Memory: Sound Shop normally only allows about 2/3 of the available memory to record a sound. This allows working room for the clipboard. If you need the maximum amount of memory to record a sound, or edit a large file, choosing this function will set the available memory to the maximum size.

Disable/Enable Cautions: Usually Sound Shop tries to remind you about the consequences of certain actions. If you would rather not bother with the occasional reminder messages, choosing "Disable Cautions" will temporarily disable them. (When loading a new file, you *will* still be prompted with a reminder if you have not saved an altered file, or you've made a new recording.) This will remain in effect as long as you are in Sound Shop.

After you select "Disable Cautions," it will change to "Enable Cautions." Pick "Enable Cautions" to turn the caution messages back on.

Status Displays

The Sound Shop screen display includes some important numbers about your sound.

Memory Status: This is the total amount of system memory available for a sound. The example screen shows a total of 819200 bytes. At a given record rate, this corresponds to a specific amount of recording time (79 seconds, in this example). Changing the slider bar in the Record Rate control will change the maximum record time displayed for the current amount of available memory.

This status box also displays the amount of memory currently used by the clipboard. Remember that your sound sample and the clipboard share a common amount of available memory. Increasing the size of one decreases the amount of memory available for the other.

Buffer Status: At startup, this indicates the amount of memory available for loading a file or recording a sound. It also translates this into a length (in seconds) for loading or recording a sound. If you have loaded a file or made a recording, this will show the amount of memory for that sample. In that case, the record time will still indicate the maximum time available, but the play time value will show you the actual time represented by your recording.

The play time shown also depends on the playback rate. The playback rate is controlled by the slider control below the Buffer Status window.

If you have anything on the clipboard, the play time of the clipboard data is also indicated in this status box.

Editor Status: This window indicates the magnitude (high/low points) of the sound wave at the point selected. This is useful for cutting, copying, and pasting sound segments from one point in the file to another. It can help you try to avoid mismatching transition values. (That is, if the sound at the end of a segment you're about to cut is loud, try not to paste it into a silent passage unless you want the resulting "bang" effect.)

Try this: click the mouse in the sound wave window. Hold down the mouse button while you drag it to one side or the other. The selected area of the waveform is highlighted, and the Editor Status box shows the magnitude of the sound wave at the current selection point.

Record Threshold: You may want the system to wait until there is an actual sound to start recording. There will always be some background noise in the system, so the threshold setting allows you to specify how loud an incoming sound should be to trigger the recording process. When a non-zero value is used, the screen border will turn green when "Record" is clicked. This indicates that Sound Shop is waiting for a sufficiently loud sound to actually begin recording. The screen border will turn red when recording actually starts.

chapter

7

Sight 'n' Sound and Sound Browser

This chapter is dedicated to Sight 'n' Sound and Sound Browser, two utility packages included with HyperStudio that make the computing experience more fun.

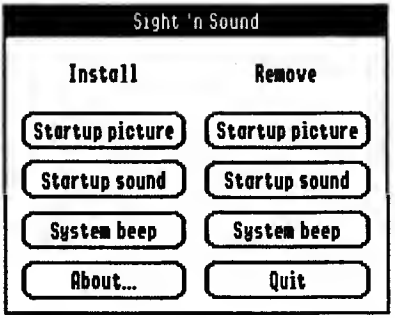
Sight 'n' Sound

Sight 'n' Sound is a fun utility. It allows you to change the startup process of your Apple IIGS. It can put a picture on the screen while the computer is starting up, or play a sound you have recorded...or both!

You can also replace the the standard Apple "beep" with a sound of your own. How about the sound of breaking glass, "oops," or a pleasant "try again?"

Sight 'n' Sound is easy to use. Click the Sight 'n' Sound icon in the HyperStudio Home Card, or launch it from the Finder or any other program launcher.

When you launch Sight 'n Sound, the menu will appear.



Let's look at each option:

Startup Picture: Click this to see the standard file dialog. Locate the picture that you wish to load, and click "Open." You can use any super hi-res picture, in 320- or 640-mode.

Once you have selected a file, a miniature image of the picture will appear. If it's not the one you want to use, click "No, pick another." Otherwise, click "Yes, install."

Now you need to tell Sight 'n Sound on which disk you want to put the picture. The disk that you place the picture on must be a bootable GS/OS disk with a system folder. You can use the "Next Device" button to scan each drive you have connected, or the "Re-Check Device" to re-examine a disk drive after you've put in a new disk.

When the name of the desired target is on the screen, click the "Install" button, and your startup picture will be installed. If a startup picture already exists on that disk, you will be asked if you want it replaced.

Note: there are actually two Sight 'n' Sound programs on the `/HS.Sounds` disk. `SNS.320` adds 320-mode Super Hi-Res graphics to a disk, while `SNS.640` does the same for 640-mode graphics. The graphics from the `/HS.Art` disk are all 640-mode, and screens from programs such as PaintWorks Gold and Platinum Paint are usually 320-mode.

Startup Sound: This is similar to the "Startup picture" function, above: it will install a Sound Shop sound on your disk that will be played when the computer starts. A sound is selected just like any other file. After you have selected it, you can play it to be sure it's the sound you want. If it is, click "Yes, install." This takes you to the same disk selection dialog box used for installing a startup picture.

Note that, if your sound sample takes longer to play than the time it takes to start up your disk, the sample will be stopped early as your startup application is loaded. The time available varies according to the type of disk drive you have, and what software is on your disk. Some customizing of the startup sound may be desired for the best effect.

System Beep: This is to replace the usual "beep" in Applesoft BASIC and other programs with a sound of your choice. The process is identical to adding a startup sound. Your custom sound will only be heard when the Apple IIGS system beep is used.

Removing Startup Files: Any or all of the files just added can be removed—simply click the "Remove" buttons, on the right of the Sight 'n' Sound function box. In each case, all that is displayed is a dialog box for selecting a disk. For each device, Sight 'n' Sound will tell you if a startup file is on that disk.

File Usage: Sight 'n' Sound adds the following files in the `System.Setup` folder of the target disk. (The `System.Setup` folder is in the `System` folder, present on any GS/OS startup disk.)

For a startup picture:

`SNS.SP`	This is the program that loads the startup picture.
`SNS.SP.PIC`	The picture to be loaded, with a special auxtype.

For a startup sound:

`SNS.SS`	This is the program that loads the startup sound.
`SNS.SS.SND`	The sound to be loaded.

For a new system beep:

`SNS.SB`	This is the program that loads the system beep.
`SNS.SB.SND`	The sound to be loaded.

The files `SNS.SP`, `SNS.SS`, and `SNS.SB` must be on the disk in the same directory as the Sight 'n' Sound utility for an installation to work properly. If the files are not found, a message will be displayed when the installation is attempted. The Sight 'n' Sound utility *does not* have to be on your own disk for the startup functions to work.

If you have a program that lets you change the order of files in a directory (such as ProSel), you can get a longer picture display time and a longer sound play time if you move the files `SNS.SP` and `SNS.SS` to the top of the `System.Setup` folder. (`SNS.SP` should not be moved to a position before `CDEV.Init` in the `System.Setup` folder.)

Sound Browser

The Sound Browser is a utility for quickly looking (listening) through sound files on a disk. It presents the usual file interface to choose the files you're interested in, and only Sound Shop-compatible files are displayed. After a sound is loaded, it is immediately played. After playing, you can choose another sound to listen to.

Some sound samples can be long. A slider control lets you adjust the amount of each sound file to be played. This makes it easier to "browse" through many samples when you're seeking something inparticular.

When you're done browsing, click "Cancel" to return to the HyperStudio Home Card, Finder, or any other program selector.

appendix

A

Tool Reference

This chapter provides you wish a quick reference of all of the paint tools and the special keys you can press when using them to produce different effects.

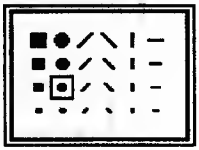
Tool Reference

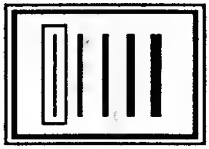
This table is a quick reference for all the paint tools and the keys you can press when using them to cause them to do different things.

When working with the paint tools, it may help to tear the Tools menu off and keep it on the screen someplace. To do this, drag down and through the side or bottom of the menu. Drag it by the top bar to reposition it, or click the square in the upper left corner to get rid of it.

Note that the Tools menu must be "torn off" to double-click one of its icons.

Icon	*Name*	*Double Clicking:*	*Modifiers:*
	Selector tool or selection rectangle	Everything is selected.	Hold the Apple key down while dragging to resize the selected object.
	Lasso		Hold the Option key down while dragging to create a copy without removing the original. If Draw Multiple is checked while Option-dragging, multiple copies will trail behind the mouse path.
	Pencil	Zooms into the "Fat Bits" mode.	Select the color or pattern from the Colors menu. Hold the Shift key down while dragging for perfectly straight horizontal or vertical lines.

	Paintbrush	Allows you to pick a new brush shape. 	Select the color or pattern from the Colors menu. Select "Brush Shapes" from the Options menu to change the brush shape. Select "Draw Multiple" from the Options menu to draw repeat images as you drag the mouse. Select "Draw Centered" from the Options menu to draw from the center instead of the end. Hold the Shift key down while dragging for perfectly straight horizontal or vertical lines.
	Eraser	Erases the entire screen.	Select "Background Color..." from the Options menu to set the color or pattern to erase with. Hold the shift key down while dragging for perfectly straight horizontal or vertical lines.

	Line tool	Allows you to specify a new line thickness. 	Select "Line size..." from the Options menu to change the line thickness. Select the color or pattern from the Colors menu. Select "Draw Multiple" from the Options menu to draw repeat images as you drag the mouse. Select "Draw Centered" from the Options menu to draw from the center instead of the end. Hold the Shift key down while dragging to constrain the tool to 45° increments.
	Spraypaint brush	No action.	Hold the Shift key down while dragging for perfectly straight horizontal or vertical lines.

	Rectangle	Toggles the drawing mode: filled or empty.	Select the color or pattern from the Colors menu. Select "Draw Filled" from the Options menu to draw solid shapes as you drag the mouse. Select "Draw Multiple" from the Options menu to draw repeat images as you drag the mouse. Select "Draw Centered" from the Options menu to draw from the center instead of a corner. Hold the Shift key down while dragging to draw squares instead of rectangles.
	Rounded rectangle	Toggles the drawing mode: filled or empty.	Select the color or pattern from the Colors menu. Select "Draw Filled" from the Options menu to draw solid shapes as you drag the mouse. Select "Draw Multiple" from the Options menu to draw repeat images as you drag the mouse. Select "Draw Centered" from the Options menu to draw from the center instead of a corner. Hold the Shift key down while dragging to draw rounded squares instead of rounded rectangles.
	Fill	No action.	No change.

	Oval	Toggles the drawing mode: filled or empty.	Select the color or pattern from the Colors menu. Select "Draw Filled" from the Options menu to draw solid shapes as you drag the mouse. Select "Draw Multiple" from the Options menu to draw repeat images as you drag the mouse. Select "Draw Centered" from the Options menu to draw from the center instead of a corner. Hold the Shift key down while dragging to draw circles instead of ovals.
	Freehand shape	Toggles the drawing mode: filled or empty.	Select the color or pattern from the Colors menu. Select "Draw Filled" from the Options menu to draw solid shapes as you drag the mouse.
	Text tool	Calls up the font selections.	Select the text style and color from the Options Menu.
	Magnifying glass	Zooms into the "Fat Bits" mode for a close-up view.	Hold down the Apple key and click to select the color under the cursor. Hold down the Option key and drag the mouse to scroll the screen.
	Polygon	Toggles the drawing mode: filled or empty.	Select the color or pattern from the Colors menu. Select "Draw Filled" from the Options menu to draw solid shapes as you drag the mouse.

appendix

B

SimpleScript Reference

In this appendix, you'll find convenient reference tables for working with SimpleScript, the language supplied with HyperStudio. Included is a command summary, ASCII chart, a table of screen and border colors, and a list of transitions.

SimpleScript Command Summary

name	*menu*	*usage*
About SimpleScript	Apple	—
Ask choice	Objects	Ask choice *expression1* with *variable*, *expression2* [, *expression3*, ..., *expression12*]
Ask filename	Objects	Ask filename *expression* with *variable*
Ask question	Objects	Ask question *expression* with *variable*
Auto indent	Apple	—
Beep	Objects	Beep
Calculator	Variables	*variable* = *expression*
Clear	Edit	—
Combine	Variables	*variable* = *expression1* & *expression2* [& *expression3*]
Copy	Edit	—
Cut	Edit	—
Debug on	Edit	—
Decrement	Variables	Decrement *variable*
Delete object	Objects	Delete *type expression1* [on card *expression2*]
Delete text	Variables	Delete character *expression* [to *expression*] of *variable*
Draw box	Screen	Draw box from *x1,y1* to *x2,y2*
Draw dot	Screen	Draw dot at *x1,y1*
Draw line	Screen	Draw line from *x1,y1* to *x2, y2*
Draw oval	Screen	Draw oval from *x1,y1* to *x2, y2*
Draw text	Screen	Draw text *variable* [;/:]
Else	Structure	Else *statement*
End If	Structure	End If
End of script	Structure	End of script
End Repeat	Structure	End Repeat
Erase file	Variables	Erase *pathname* [with *variable*]
Export script	File	—
Fill flag	Screen	Fill flag is *On/Off*
Find character	Variables	Find ASCII character *expression1* in *expression2* with *expression3*
Find line	Edit	—
For	Structure	For *variable* = *expression1* to *expression2*

Get ASCII	Variables	Get ASCII of *expression1* with *expression2*
Get border color	Info	Get border color into *variable*
Get card number	Info	Get card number into *variable*
Get event	Info	Get event into *variable1*, *variable2*
Get field	Objects	Get field *expression* into *variable*
Get filename	Objects	Get filename *expression* with *variable*
Get length	Variables	Get length of *expression* into *variable*
Get max possible	Info	Get max possible into *variable*
Get menu mode	Info	Get menu mode into *variable*
Get number of...	Info	Get number of *type* into *variable*
Get rect	Objects	Get rect of *type expression1* [on card *expression2*] into *x1, y2, x2, y2*
Get score	Info	Get score into *variable*
Get stack name	Info	Get stack name into *variable*
Get text	Variables	*variable* = character *expression1* [to *expression2*] of *expression3*
Get time and date	Info	Get time and date into *variable1, variable2*
Get version	Info	Get version into *variable*
Global	Variables	Global *variable* [,*variable*]
Go to button	Structure	Go to button *variable*
Go to card	Structure	Go to card *variable*
Go to remark	Structure	Go to remark *variable*
Gosub button	Structure	Go to a card *expression*
Gosub proc	Structure	Gosub proc *expression*
Hide cursor	Screen	Hide cursor
Hide object	Objects	Hide *type expression1* [on card *expression2*]
If	Structure	If *expression* ¶ *statement* ¶ End If
Import script	File	—
Increment	Variables	Increment *variable*
Insert text	Variables	Insert *expression1* into *variable* at character *expression2*

Make character	Variables	Make character from ASCII of *expression* into *variable*
Move to	Screen	Move to *x, y*
NBA	Objects	NBA *name* [*expression1, ..., expression4*] [into *variable1, ..., variable4*]
New script	File	—
Next	Structure	Next [*variable*]
Page setup	File	—
Paste	Edit	—
Pause	Structure	Pause *expression ticks/seconds*
Play sound	Objects	Play sound *pathname*
Print	File	—
Procedure	Structure	Procedure *expression*
Quit	File	—
Read file	Variables	Read file *pathname* into *variable1* [with *variable2*]
Read mouse	Screen	Read mouse position into *x, y*
Redraw object	Objects	Redraw *type expression1* [on card *expression2*]
Refresh screen	Screen	Refresh screen
Remark	Edit	—
Repeat until	Structure	Repeat until *expression*
Replace text	Variables	Replace *expression1* with *expression2* in *variable*
Return	Structure	Return
Save and quit	File	—
Select all	Edit	—
Set border color	Screen	Set border color to *variable*
Set field	Objects	Set field *expression1* from *expression2*
Set font	Screen	Set font to *fontname size* [*style*]
Set line color	Screen	Set line color to *expression*
Set line size	Screen	Set line size to *expression*
Set next transition	Screen	Set next transition to *transition*
Set pen mode	Screen	Pen mode is *XOR/Bic/Copy/And*
Set rect	Objects	Set rect of *type expression1* [on card *expression2*] from *x1, y2, x2, y2*
Set text color	Screen	Set text color to *expression*
Set variables	Variables	*variable = expression*
Show cursor	Screen	Show cursor

Show message	Objects	Show message *expression* [with icon *icon*]
Show object	Objects	Show *type expression1* [on card *expression2*]
Sort on text field	Objects	Sort on text field *expression* in *ascending/descending* order
Swap cards	Objects	Swap card *expression1* with *expression2*
Swap	Variables	Swap variables *variable* with *variable*
Undo	Edit	—
Write file	Variables	Write file *pathname* from *variable1* [with *variable2*]

SimpleScript Reserved Words

This is a list of words that cannot be used as the beginning of (or all of) a variable name.

ALL
AND
AS
AT
BEEP
BIC
BOLD
BUTTON
BY
CARD
CARDS
CHARACTER
COPY
COS
DEBUG
DESCENDING
DECREMENT
DELETE
DO
DRAW
ELSE
END
ERASE
EXP
FILL
FIND
FOR
FROM
GET
GLOBAL
GO
GRAPHIC
HIDE
ICON
IF
IN
ITALIC
ITEM
LINE
LN
LOG2
MAKE
MOVE
NBA
NEXT
NOTHING
OF
ON
OR
OUTLINE
PAUSE
PEN
PLAY
PROCEDURE
RANDOM
READ
REDRAW
REFRESH
REMARK
REPEAT
REPLACE
RETURN
ROOT
ROUND
SECONDS
SET
SHADOW
SHOW
SIN
SORT
SQR
SWAP
TAN
TEXT
TICKS
TO
TRUNC
UNDERLINE
WITH
WORD
WRITE
XOR

ASCII Chart

This chart shows the ASCII value (numerical representation) of the SimpleScript character set. It is useful for the `Make character` and `Get ASCII` commands.

ASCII value	*character*	*ASCII value*	*character*	*ASCII value*	*character*
32	*(space)*	64	@	96	`
33	!	65	A	97	a
34	"	66	B	98	b
35	#	67	C	99	c
36	$	68	D	100	d
37	%	69	E	101	e
38	&	70	F	102	f
39	'	71	G	103	g
40	(	72	H	104	h
41	)	73	I	105	i
42	*	74	J	106	j
43	+	75	K	107	k
44	,	76	L	108	l
45	-	77	M	109	m
46	.	78	N	110	n
47	/	79	O	111	o
48	0	80	P	112	p
49	1	81	Q	113	q
50	2	82	R	114	r
51	3	83	S	115	s
52	4	84	T	116	t
53	5	85	U	117	u
54	6	86	V	118	v
55	7	87	W	119	w
56	8	88	X	120	x
57	9	89	Y	121	y
58	:	90	Z	122	z
59	;	91	[	123	{
60	<	92	\	124	|
61	=	93	]	125	}
62	>	94	^	126	~
63	?	95	_	127	

Screen and Border Colors and Transitions

These tables may be useful when programming with SimpleScript or other HyperStudio languages.

SimpleScript doesn't just use "red," "blue" or other color names because the actual color may vary if you are using custom colors on a card. However, if you like, you may set your own variables: for example,

```
Red = 15
Set text color to red
```

Screen Colors

Note that these colors are based on the standard 640-mode dithered palette that HyperStudio uses as a default with a new stack; if the picture's color set changes, colors may not match their values in this table (for example, color 4 may no longer be red but some shade of gray).

0 black	6 orange	12 gray #2
1 dark blue	7 pink	13 per. blue
2 olive	8 dark green	14 yellow
3 gray #1	9 turquoise	15 white
4 red	10 neon green	
5 purple	11 light green	

Border Colors

0 black	6 medium blue	12 light green
1 deep red	7 light blue	13 yellow
2 dark blue	8 brown	14 aquamarine
3 purple	9 orange	15 white
4 dark green	10 light gray	
5 dark gray	11 pink	

Built-in Transitions

Barn close	Diamond dissolve	Mouth open
Barn open	Dissolve	Rain
Bars	Fade to black	Razor left
Blocks	Fade to white	Razor right
Border color fade	Fastest	Right to left
Bottom to top	Iris close	Top to bottom
Bow ties	Iris open	Venetian blinds
Diagonal left	Left to right	Zoom in
Diagonal right	Mouth close	Zoom out

appendix

C

Copying a Disk

This appendix details exactly how to copy a diskette, so that you may back up your HyperStudio original disks. Additionally, it explains how to make a data disk.

Making a Data Disk

To make a data disk, you will need your Apple IIGS System Disk. We will be using the Finder; it's the program that you automatically go into when you start up the system disk. (You can tell you're in the Finder when there's a little picture of a trash can in the bottom right corner of the screen.)

- Be sure you have a blank 3.5" disk available. (It does not need to be formatted.) Start up (boot) your computer with the Apple IIGS System Disk, which came with your computer.
- Once you're in the Finder (remember, look for the trash can in the corner), insert your blank 3.5" disk.

  If you have two drives, insert the blank disk in the second drive; otherwise, remove your System Disk from the drive and insert your blank disk. You may be asked to reinsert your System Disk at a later time—simply follow the prompts as the computer directs you.
- *If the disk is not formatted,* the Finder will display a dialog box with the message, "GS/OS can't read this disk. Do you want to initialize it?" Respond by clicking the "Continue" button.

  You'll get a second dialog box, with two lists in the middle of the dialog box and two buttons at the bottom of the box: "Cancel" and "Initialize". Check that the left list has "ProDOS" highlighted and the right list has "800K" or "800K 2:1" highlighted, then click "Initialize" (or press the Return key). The computer will proceed to format the disk, which takes about a minute. When the computer is done, you've got a new blank data disk.
- *If your disk has already been initialized,* you'll see it appear on the desktop as a small picture of a disk with the name of the disk directly below it. Verify that the disk does not contain any data that you want to keep—we'll be erasing it in just a moment.

  Now, click the disk once, to cause it to highlight, and then select "Erase..." from the Disk menu (at the top of the screen). A dialog box will appear with the name of the disk at the top and two list, on the right and the left. The bottom of the dialog box will have two buttons: "Cancel" and "Erase."

  You need to give the disk a name before you can erase it. You should name it something like "Data.Disk" or "HS.Work" or something similar, that indicates to you that it is your data disk. (You can also name it after yourself, if you'd like!) To give the disk a name, just type it in.

  Finally, check that the left list has "ProDOS" highlighted and the right list has "800K" or "800K 2:1" highlighted, and click "Erase." The computer will quickly erase your disk, and then it will appear on the desktop. When it's done, you're done, and you've got a new blank data disk.
- *If the disk appears on the desktop with a small picture of a padlock next to the name,* it means that it is disk is locked. Press the eject button the drive, and look at the disk. Be sure it's a blank disk (or some other disk that you can copy over) and not your original HyperStudio disk! (You always want to keep your original HyperStudio disk locked.) Slide the disk write-protect switch to the lower position, so that it covers the hole in the

corner of the diskette, and and put it back in the drive. The padlock next to the name should disappear automatically. Remember, by unlocking the disk, you're paving the way to erasing it, so be sure that the disk doesn't contain anything you wish to keep.

Copying a Disk

The following steps will explain how to copy your original HyperStudio disks.

- Check your original HyperStudio disks and verify that the write protect switch is in the upper position on all of them (so that you can see through the hole in the comer of the disks). This insures that there is *no way* for you to destroy your original disks—it's impossible for the computer disk drive to write to a disk that has the write protect switch in the upper position.
- Make as many blank data disks as you need to make copies. Since the HyperStudio package is six disks, you should make six data disks. Instructions for making a data disk are at the beginning of this appendix.
- If you haven't already, boot your Apple IIGS System Disk, which came with your computer. When the computer is finished booting, you will be in the Finder. (You can tell you're in the Finder because of a little picture of a trash can in the corner of the screen.) The Finder is the program that we are going to use to copy the disks.
- Press the eject button on the disk drive to remove your blank 3.5" disk and insert your original HyperStudio disk. If you have two disk drives, keep the System Disk in the drive and simply insert your original HyperStudio disk in the second drive.
- The HyperStudio disk will appear on the desktop. If the HyperStudio disk doesn't have a small picture of a padlock next to the name "HyperStudio," eject the disk and be sure the write protect switch is set to the upper position, so you can see through the hole in the disk. The padlock means that the disk is "locked," and that the computer can't write anything to it—which is how you want all of your original disks to appear to the computer.

(continued on following page)

Note: be sure to read the following steps and be certain that you understand them before doing them.

- Eject your original HyperStudio disk by pressing the disk eject button on the disk drive. The icon for the HyperStudio disk will remain on the screen, but it will be dimmed. Insert one of your blank data disks.
- Drag the dimmed HyperStudio disk icon *on top of* the icon for the blank data disk, and be sure the data disk's icon lights up. (Think of it as "dropping" the HyperStudio disk on top of the data disk.) If the data disk icon doesn't light up when you move the HyperStudio disk icon on top of it, be sure you've got the tip of the arrow pointing to the data disk, and remember to hold the mouse button down!
- Release the mouse button. You will get a dialog box with the message, "Completely replace the contents of 'Untitled' [or whatever your data disk is named] with the contents of 'HyperStudio'?" Verify that this is correct, and then click the "OK" button, or press Return.
- Follow the prompts. The Finder will ask for the data disk and for the HyperStudio disk. When it says to insert one of them, simply press the eject button to remove the disk in the drive and insert the disk that the Finder needs. This will happen several times. As the Finder copies the disk, you can tell how far it has gone via the red 'thermometer' indicator.
- When the disk copy is completed, the last step is to change the name of the data disk disk to the name of the disk you just copied on top of it: "HyperStudio". Do this by first dragging the icon of the "HyperStudio" disk to the trash. (You do this because you can't have two disks of the same name on the desktop, so we have to make the Finder 'forget' about the original HyperStudio disk icon before we can change the name of another disk to "HyperStudio".) Note that dragging a *disk* icon to the trash can will not erase it. It only causes the Finder to forget about it. After you've dragged the "HyperStudio" disk icon to the trash, it will disappear from the screen completely.
- Click the data disk's icon once to highlight it. Type the word "HyperStudio", then press Return. (If you make a mistake, click somewhere *away* from the disk icon to deselect it, and then repeat this step again.)

That's it! Now you have a copy of the HyperStudio disk. Follow these same steps to copy any other normal (that is, non-copy-protected) 3.5" disk, including the remainder of the disks in the HyperStudio package.

appendix

D

Sample NBAs, Extras, and Transitions

This appendix documents the sample New Button Actions (NBAs), Extras, and Transitions that come with HyperStudio.

New Button Actions (NBAs)

New Button Actions (NBAs) are special programs that extend the functionality of HyperStudio by giving an buttons entirely new action. As the title indicates, a New Button Action is HyperStudio's way of providing endless possibilities in future expansion of the system.

HyperStudio comes with a number of New Button Actions (NBAs) built into the program, to make your stacks do new and exciting things. They are documented below. You may find that one (or a few) of the NBAs below could take the place of loading an entire language into HyperStudio.

Note that some of the NBAs listed below are built into HyperStudio, and others are stored on the HyperStudio disk (in the NBAs folder). The NBAs stored on disk can be accessed by clicking the "Disk library..." button, on the New Button Actions screen.

Port

by Ken Kashmarek

With the Port NBA, you can send ASCII data out of the printer or modem port. This is most useful to send commands to a laserdisc player attached to the modem port, but can also be used to send data to a modem or printer.

To use the NBA, simply select it from the NBA list, click "Use NBA...", and select the following options:

- **Port #1:** Selecting this will instruct the Port NBA to send data out of the printer port.
- **Port #2:** Selecting this will instruct the Port NBA to send data out of the modem port.
- **Reply NO:** If you mark this option, the NBA will not wait for a response from the device attached to the port.
- **Reply YES:** Marking this option will require that the device attached to the port send back a response before the NBA will proceed.

The commands that you select will appear in the NBA message box (directly below the NBA list). You can then enter the commands that you wish to pass to the device attached to the port you specified by simply typing them in the message. For example, after clicking the "Port #2" and "Reply NO" radio buttons in the Port NBA dialog box and then clicking "OK," the NBA message box will look like this:

```
Port #2
Reply NO
```

You can proceed to enter text to be passed through the port below "Reply NO".

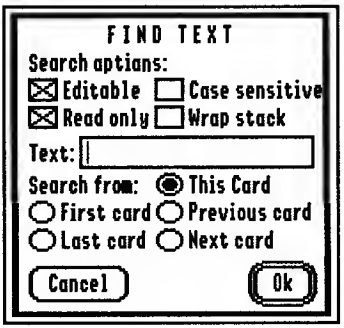
As an example, to start a laserdisc that is attached to the modem port and not wait for a response from the player, the message box might read:

```
Port #2
Reply NO
PL
```

Find

by Ken Kashmarek

The Find NBA allows your button to search through all of text items in a stack for a specified string. You have several options available in the NBA, as discussed below:



☒ **Editable:** Marking this will tell the NBA that it may search in editable text items.

☒ **Read only:** Marking this will tell the NBA that it may search in read-only text fields. By marking this and the "Editable" checkbox, the NBA will search all text items.

☒ **Case sensitive:** By marking "Case sensitive," the NBA will require that the text you search for match exactly with what it finds. If you mark this checkbox, a search for "Wow" will not find "WOW" or "woW".

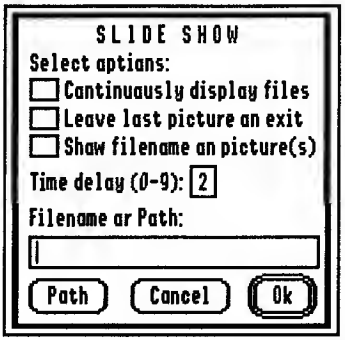
☒ **Wrap stack:** If you mark this checkbox, when the Find NBA reaches the end of the stack it's searching, it will continue the search at the beginning of the stack.

After entering the text you wish the NBA to search for, and specifying where you want the search to start (with the first card in the stack, the last card, the previous card, the next card, or the card where the button is located), click "OK".

Slide Show

by Ken Kashmarek

The Slide Show NBA allows you to start a slide show with the click of a button. You may display a single picture, or all of the pictures in a directory. There are a number of options:



☒ **Continuously display files:** If you mark this option, the slide show will continually run: when it reaches the end of a directory, it will start again from the top.

☒ **Leave last picture on exit:** By marking this option, when the slide show finishes, the last screen displayed will remain on the card, instead the card's background being redrawn.

☒ **Show filename on picture(s):** Marking this option will display the filename of the current picture in the lower-right corner of the screen.

The "Time Delay" edit line allows you to specify how quickly the slide show runs. Smaller numbers make the show run faster.

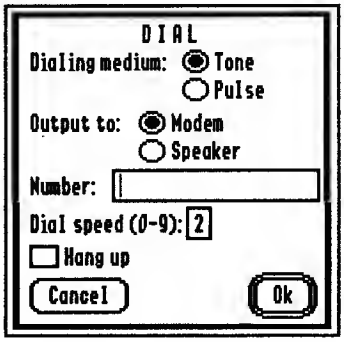
The "Filename or Path" edit line allows you to specify a single filename (if you wish to display a single picture) or the path to the directory (folder) you where you wish to display all of the pictures enclosed within. The easiest way to set this edit line is to click the "Path" button, which will cause the standard file dialog to appear. You may specify a single picture by selecting it and clicking "Open," or you can specify an entire disk directory by selecting the directory, clicking "Open" to open the folder, then clicking "Cancel".

If you wish to display all pictures in the directory with the stack, simply enter a single space in the "Filename or Path:" edit line.

Dial

by Ken Kashmarek

The Dial NBA allows the user to enter a phone number to dial "live" in a stack, or, if a range of text is selected in a text item when this NBA is called, it is assumed that the selected text is a phone number and the number will be dialed.



◉ **Tone:** Selecting this option will cause the NBA to dial in tone mode. Note that you can dial in tone mode out of the speaker as well as the modem.

◉ **Pulse:** Selecting this option will cause the NBA to dial in "pulse" mode. Pulse dialing can only happen from the modem.

◉ **Modem:** By default, this NBA will dial the phone via a modem attached to the modem port.

◉ **Speaker:** Normally, this NBA dials via a modem attached to the modem port. By marking this option, the NBA will dial using tone dialing through the Apple IIGS speaker. If you have an amplified speaker attached to your computer, you can hold the phone up to the speaker and the NBA will dial the phone!

☒ **Hang up:** By marking this option, the Dial NBA will hang up the phone (through the modem) once a connection has been made. This is only necessary when dialing through the modem.

Dial speed: You may specify the dialing speed by entering a number in the "Dial speed" edit line; 0 is the fastest speed and 9 is the slowest.

Date

by Ken Kashmarek

The Date NBA will search text items in a stack for the current date (as reported by the Apple IIGS' system clock). This is particularly useful for a datebook stack.

☒ **Editable:** Marking this will tell the NBA that it may search in editable text items.

☒ **Read only:** Marking this will tell the NBA that it may search in read-only text items. By marking this and the "Editable" checkbox, the NBA will search all text items.

☒ **Wrap stack:** If you mark this checkbox, when the Date NBA reaches the end of the stack it's searching, it will continue the search at the beginning of the stack.

The six "Search from" options simply control where the date search begins (from the current card, the previous card, the next card, the first card in the stack, or the last card).

◉ **MM/DD:** By marking this option, you specify that the Date NBA should look for the date in a month/day format (without being concerned about the year).

◉ **MM/DD/YY:** This option is similar to the one above, except it searches the stack for a date in month/day/year format.

Note that the format of the date in the stack must me the same as the format you're searching for (MM/DD or MM/DD/YY), without spaces or leading zeros. (I.e. the date "10/08" won't be found even if you're searching in MM/DD format, but "10/8" will.)

Auto Record

by Dave Klimas

This NBA allows you to record a sound directly with a button. This is used in stacks such as foreign language, speech therapy, and others, where you want the user to record their voice without having to know anything about how to create a button with HyperStudio.

To use the Auto-Record NBA, click "Disk Library..." at the New Button Actions dialog box, then select the "AutoRecord" file. Finally, click "Use NBA...".

(continued)

A text item will appear below the NBA list, which is where you enter information for the Auto Record NBA. This is sometimes called the "command line". A typical line might look like:

```
+R2 +T10 +D20 my.sound
```

The options are:

+R*x* How many seconds of sound you want to record. In the example above, two seconds of sound will be recorded. If you don't specify a value, the NBA will record four seconds worth of sound. You should give some thought to picking an appropriate value for the record time: if you pick a value that is too short, the person's words may be cut off. If you a pick a value that is too large, then you'll be taking up extra disk space, and in addition, you'll be recording extra space at the end of their voice.

+T*x* This is a threshold value of how loud of a sound will start the actual recording process. Like the HyperStudio Tape deck, the Auto Record NBA first turns the screen border green, meaning it is waiting for you to actually start speaking. When it hears your voice, the screen border turns red, indicating that it is actively recording. If your computer is electrically noisy, or for any other reason you want to change the sensitivity of the voice activation, you can enter a number between 0 and 127 after the characters "+T" in the command line.A value of "10" is reasonable, and is shown in the example.

Values too high will prevent the NBA from ever starting the recording process, or may also "clip" the first part of what you are trying to record. If you don't specify a value, "32" is used (this is a pretty high value, even for noisy machines).

+D*x* Delay for time-out. If the threshold value is too high, or the person never starts speaking, the NBA might wait forever. The delay value set how long (in seconds) you want the green border to remain. If the person doesn't speak for a period of time that exceeds your delay value, then AutoRecord will automatically return control back to the stack. In the example, the user is given 20 seconds to start speaking once they click the button and the border turns green. If you don't specify this value, Auto Record will wait a maximum of four seconds, and then "time out."

`filename` In the example, "my.sound" is the name of the file that will be written to disk with the recorded sound. All of the other values are optional except for this; a filename must always be included on the command line for the NBA.

Load Font

by Ken Kashmarek

With this NBA, you may load fonts in your stack "on the fly"—as soon as this NBA is triggered and passed the pathname of a font file (or files), the font(s) will be loaded and installed. This means that you may immediately start using the fonts in text fields, whereas they would otherwise have to be installed in the user's system fonts folder.

To use the NBA, simply select it from the NBA list, and click "Use NBA...". A standard file dialog will appear, allowing you to select the fonts you want to load by opening them. After you open a font, the standard file dialog will re-appear, allowing you specify more fonts to open. Continue like this until you have selected all that you wish to be loaded, then click "Cancel." The fonts will be loaded when the user clicks on the button triggering the NBA.

The Load Font NBA is sensitive to trailing spaces and blank lines after the name of a font. That is, if you specify the font to be loaded as "Shaston 16 " (note the extra space at the end of the name), the font won't be properly located by Load Font. The easiest way to check for extra spaces or blank lines in the Load Font command box is to just click the mouse in the blank space near the bottom of the window, and see where the cursor appears. If it appears at the end of the typed text, you're all set. If it appears after the word, press the Delete key until has moved back to the very end of the line of text, then click "Done" in the dialog box.

Also, if you leave the name of the font that you wish the Load Font NBA to load completely blank, then the NBA will prompt the user for a font to load with the standard file dialog box. Although this may not seem to be very practical for the novice stack user, you may find it personally useful to create a stack which could load fonts from other disks "on the fly," as it were.

Card-O-Matic

by Michael O'Keefe

Card-O-Matic is an NBA that allows you to work with entire cards from a button. With it, you may perform a number of card-related functions.

(continued)

To use Card-O-Matic, simply select it from the NBA list, click "Use NBA...", and then enter one of the following commands:

First, to move to the first card in the stack.
Last, to move to the last card in the stack.
Next, to move to the next card in the stack.
Previous, to move to the previous card in the stack.
Back, to move to the last card that the user was on.
Cut, to cut the current card from the stack, and move it to the clipboard.
Copy, to copy the current card to the clipboard.
Paste, to insert the current card in the clipboard into the stack.
Delete, to completely delete the current card.
New, to create a new card. (This is the same as pressing ⌘-N.)
MoveTo, to move to a specific card. Follow the word "MoveTo" with a card ID or name.

You may string together several commands by simply placing them on separate lines.

Sort Cards

by Michael O'Keefe

With the Sort Cards NBA, you can sort cards based on a key text item.

To use the Sort Cards NBA, select "Sort Cards" from the NBA list, and click "Use NBA..." Enter the text item ID (or name) that you wish to sort on, followed by a "0" for an ascending sort or "1" for a descending sort.

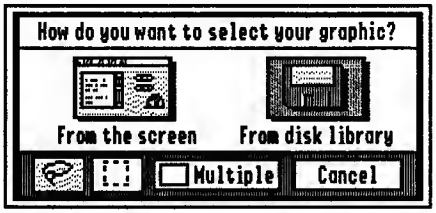
Note that only cards which contain the text item you specify will be moved in the stack; if you have a stack of ten cards but only card 4, 6, 7, and 8 contain the text item that you specify, those four cards will be sorted and placed back in the stack at positions 4, 6, 7, and 8.

Animator

by Steve Allen

Animator allows you to select an area of the screen to animate. Unlike HyperStudio's built-in animation, however, which is based on the traditional "cel" method (displaying several different frames in rapid succession to give the illusion of movement), when using Animator, you specify a path for a still object to follow.

To use Animator, select it from the NBA menu and then click "Use NBA..." You will be returned to the current card, and the dialog box shown to the right will appear, asking where you wish to get the animated graphic. You can animate something currently on the screen, or select something from a clip-art file on disk.



If you select "From disk library," the standard file dialog will appear, prompting you for a standard Super Hi-Res screen, which will then be loaded.

Additionally, you may pick the tool you wish to use when selecting a graphic to be animated: either the selector tool or the lasso tool. Note that the lasso tool will always ignore the outside color, and will also "mask out" any colors inside of the image that match the outside color. (This way, if you lasso a donut, the background will show through the hole in the middle.) However, you can *un*select the "This color is invisible" option so that *all* colors will appear in the graphic image that you lasso without having the graphic image actually appear as a box, as it does with the selector tool.

If you want the selector tool to behave like the lasso tool (so that it "wraps" to the graphic image and masks the outside color), hold down the Apple key while selecting the graphic with the selector tool.

✔ Advanced User

An additional checkbox will appear in the bottom of the dialog box: "Multiple." By setting this option, the graphic object will not be erased as it follows the trail you specify. (This is similar to HyperStudio's "Draw Multiple" option.)

In either case, your cursor will have changed to either a crosshair or lasso (depending on the tool that you seelected), and you may select an area of the screen. (At this point, you may also press the Tab key to switch between the lasso and the selector tools.)

After selecting an area, you'll see the current card, and the border color will change to green. This indicates that Animator is ready to start recording your movements. You'll notice that you cursor has changed from a crosshair to the graphic image you selected.

Now, you need to determine where on the screen you wish your animation to start. Unless you tell it to do otherwise, later, any animation sequences you create with Animator will play in front of the card's background but behind any objects on the card (including graphic objects).

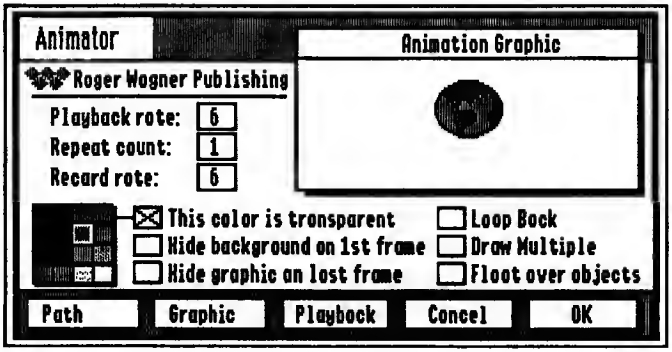
To define the animation path, hold down the mouse and drag. As soon as you hold down the mouse, the screen border will change to red, indicating that your movements are being recorded. When you are finished defining the animation path, release the mouse button. Animator's window will appear, which offers you several options for controlling your animation.

✔ Advanced User

Note that you can release the mouse button to come out of record mode, and then hold down the mouse button again to being recording a new path which is still part of the same animation. (This effect can be used to draw many stars, for example: define your graphic as a star, then simply click at random places on a background of a starry nighttime sky.)

When you are finished defining the entire animation, press Return. Animator's window will appear, which offers you several options for controlling your animation.

Playback rate: you can control the speed at which the animation plays back by changing this number. The fastest speed is 0, while 99 is the slowest. (And 99 is very, very slow!)



✔ Advanced User

Record rate: you can control the speed at which Animator records your motions. The fastest speed is 0, while 9 is the slowest. This controls how fast Animator collects data during the record mode.

This function allows for slowing down the record process for "unsteady hands," or, if the record rate is "0" (zero), a much smoother animation can be created because more frames per second is being recorded.

Repeat count: here, you may specify how many times the animation should play. Entering "0" will cause the animation to play forever, or until the user clicks the mouse or presses a key.

There are several checkboxes that allow you to control the behavior of your animation:

☒ **This color is transparent:** by marking this option and indicating a color in the color set to the left, that color will become "transparent" in the animation. The result will be that anyplace where that color appears in the animated image, the background will show through.

☒ **Hide background on 1st frame:** on a solid background, this option will clear the starting point of the animation to the background color. By having the object being animated already drawn on the card, this can make an object come "off" of a card and animate. For example, if you have a picture of a flying saucer on a solid blue background, and you start the saucer animation directly on top of the saucer painted

onto the background, marking this option will cause Animator to clear the saucer on the background after drawing the first frame of the animation.

☒ **Hide graphic on last frame:** this option causes the graphic being animated to disappear on the last frame. Normally, it remains on the card.

☒ **Loop back:** by marking this option, the animation that you have created will play completely through (as normal), and then play in reverse.

✔ **Advanced User**

After selecting "Loop back," advanced users may choose from one of three options: to toggle the "Float over objects" checkbox, to flip the graphic horizontally, or to flip the graphic vertically. Any of these options that you select will occur before the graphic makes the "return trip" on the animation path.

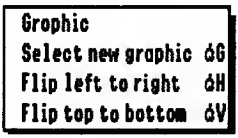
☒ **Draw Multiple:** marking this checkbox tells Animator that it should not erase the image as it is drawn—that is, a copy of the graphic should be left on the screen as it is moved along the path.

✔ **Advanced User**

☒ **Float over objects:** normally, Animator draws "underneath" of graphic objects—this way, you can give your animations a three-dimensional effect. However, with this option checked, Animator will draw on top of graphic objects, effectively making your animated object the "frontmost" thing on the screen.

Note that, even if this is *not* marked, your animation always floats over objects when examining your animation by selecting "playback" from the Animator NBA.

The "Graphic" Menu



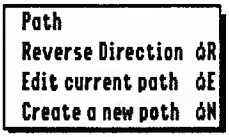
The "Graphic" menu contains items that let you work with the graphic image being animated.

Select new graphic (⌘--G): if you select this option, you can pick an entirely new graphic from the screen or disk file. (Note that this will *not* destroy or otherwise change your animaton path.)

Flip left to right (⌘--H): marking this option will flip the graphic you have selected horizontally.

Flip top to bottom (⌘--V): marking this option will flip the graphic you have selected vertically.

The "Path" Menu



Under the "Path" menu, you'll find options that offer you flexibility in working with your animation path.

Reverse direction (⌘--R): with this option marked, the animation will play in reverse order, traveling backwards along the path you have defined. This is useful if you know where you want your animation to

end. For example, if you are doing an animation that ends with a circle in the center of a box on the screen. To do this without the "Reverse direction" checkbox marked, you'd have to center the circle in the box at the end of the animation, and if you weren't precise in your first guess as to where the circle should be, the animation will include you carefully moving the circle into place.

By starting the circle in the box and then moving it out and completing the animation, then marking the "Reverse direction" checkbox, the circle will play through the animation and end up exactly in the center of the box at the end.

Edit current path (Ó--E): marking this option will flip the graphic you have selected vertically.

By choosing "Edit current path," you will be shown the current card with the animation path displayed in red. At several points along the animation path, there will be green handles. If you drag these handles, you can change the shape of the animation path.

Additionally, by positioning the mouse on the red path itself, the cursor will change to a four-headed arrow, and you can drag the entire path to a new location. (Interesting effects can be created by dragging the beginning or ending of the path off the screen!)

Holding down the Apple key and the mouse button when the cursor is *not* on any of the animation path will playback the animation with the path displayed. Holding down the Apple key while clicking on one of the points between two of the path's handles will allow you to change it's position. Holding down the Option key and clicking will allow you to add a new point to the animation path. Pressing the Delete key while editing a path will remove the most recently added endpoint of the animation path.

Pressing Apple-Space will clear the screen to white, leaving only the path on the screen. This is very useful if your path is on a busy (colorful) screen. (Press Apple-Space again to restore the normal background.)

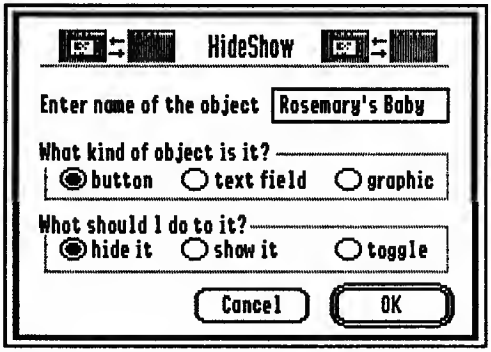
If you press Apple-R, the path's direction will be reversed. (This is similar to selecting "Reverse direction," under the "Path" menu.) Finally, pressing Escape or Apple-period will cancel any editing and restore the old animation path.

Create new path (Ó--N): selecting this will completely discard your current path and allow you to create a new path for your graphic image to follow. If you press the escape key or type Apple-period ***before the mouse is clicked,*** you will cancel this operation.

The last three buttons in the Animator NBA are "Playback," to see your animation, "Cancel," to discard your animation, and "OK," to save what you have.

Hide Show

by Michael O'Keefe

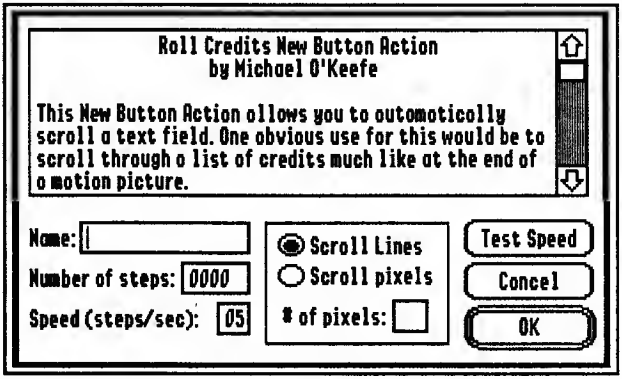


This NBA allows you to hide or show a graphic object, a button, or a text item. You can achieve impressive "special effects" by pre-hiding graphic objects, then showing them at the click of a button—possibilities include fake dialog boxes, "pop-up menus," and more.

To use the HideShow NBA, simply select it from the NBA menu and then click "Use NBA..." You will be prompted for three things: the name of the object you wish to work with, the type of object (graphic object, button, or text item), and whether you want to hide it, show it, or toggle the object (that is, hide it if it's visible, or show it if it's already hidden).

Roll Credits

by Michael O'Keefe



With the Roll Credits NBA, you can automatically scroll a specified text field at any rate. This can lend itself to interesting effects and is also very good for a "rolling credits" display, similar to the credits at the end of a movie.

To use the NBA, you need to specify the name of the text field (on the same card as the button) that you wish to scroll, and the number of times ("steps") it will scroll, and finally, the speed (in steps per second).

You can also control how smooth the scroll is by choosing one of the following options:

- **Scroll lines:** marking this option will cause the text field to scroll a line at a time.
- **Scroll pixels:** marking this option will cause the text field to scroll on a pixel-by-pixel level instead of line-by-line. If you mark this option, you should specify how many pixels (per scroll) you want the display to move. Enter this number at "# of pixels:".

To create a "rolling credits" display with an Apple II Video Overlay Card, make the background color of the text item the same as the key color, and be sure the text item does not have a border or scroll bar (but it must have the "allow scrolling" checkbox turned on!). To make the credits scroll completely on to and off of the screen, start and end the contents of the text object with several carriage returns—this will make a blank

area that will be scrolled off at the beginning (with the credits coming in from the bottom), and then a blank area at the end of the credits which will be scrolled up (with the credits going off the top).

MiniScript

by Ken Kashmarek

MiniScript is an NBA built into HyperStudio which allows you to link together a series of other NBAs without having to use SimpleScript or another language.

For example, if you have a stack and you simply want a way to hide three text fields, delete a card, and show a graphic object (for example), it saves a lot of memory to use MiniScript to tie together the HideShow and Card-O-Matic NBAs, rather than taking on the memory burden of SimpleScript for a simple ten line program.

Note that MiniScript can *only* link together other NBAs; SimpleScript has many, many more capabilities. MiniScript is a very specialized NBA that's useful only in certain situations. Additionally, only some NBAs can be used by MiniScript—for details, you should check the instructions for the specific NBA.

To use MiniScript, simply click "MiniScript" in the NBA list and then click "Use NBA..." In the text box above the "Use NBA..." button, enter the NBA commands as directed by each specific NBA. Each NBA's specific commands should each be its own line, starting with the name of the NBA. For example, you may enter:

```
Card-O-Matic New
Loadfont Kumquat.36
```

to create a new card (with the Card-O-Matic NBA) and then load the font named "kumquat.36" (with the Loadfont NBA).

Special note to users of the Master XCMD, from HyperStudio XCMDs Vol. 1: If you used the old Master XCMD system, with HyperStudio, you'll be happy to know that MiniScript works *exactly* the same way, with the additional feature that you may now build scripts directly in the MiniScript NBA window, rather than store them in a text field. Scripts stored in text files or text fields are still accessible through the old Master XCMD "`Script` <*scriptname*>" protocol.

MiniScript also supports `XCMD` .*xyz* files stored on the disk, so that you may support old XCMDs which you may have written for HyperStudio 2.1 stacks.

Under HyperStudio 3.0, the Master XCMD file "HS.XCMD" is no longer required on the System Disk (neither the 4.5k or 3k version). If your stacks only used individual `HS.XCMD` files from the `HS.Demo` disk, MiniScript supports those as well. (Those files should *not* be deleted from your stack directories.)

Note that you may use anything in the standard HyperStudio package in your own work without any additional fee, as long as you include the following credit line in a conspicuous location in stack: "Portions of this stack use HyperStudio New Button Actions copyright 1991 Roger Wagner Publishing, Inc."

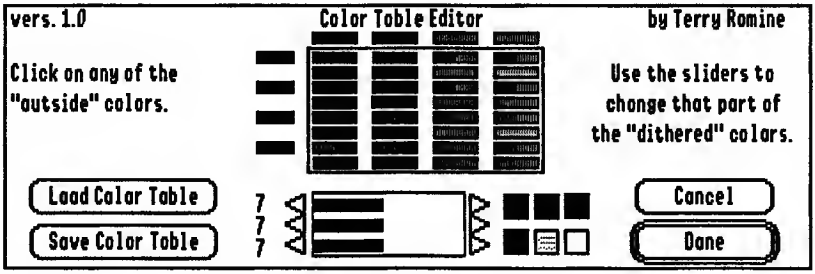
Extras

Like NBAs, a number of sample Extras are included with HyperStudio. Extras serve the purpose of small utilities, available while you are creating your stack. A button may not access an Extra; it is solely for the stack author.

Color Editor

by Terry Romine and Roger Wagner

The HyperStudio Color Editor extra allows you to edit the 640-mode dithered palette. It has been specially designed to show you exactly how modifying one color affects the other colors in the table.



Color Editor

To understand how to operate the Color Editor, you need to know a little bit about dithered colors on the 640-mode screen.

Normally, there are four colors available in 640-mode, plus black and white. These six colors are represented at the bottom of the dialog box, to the right of the "Done" button.

Additionally, they are distributed across the top and the left side of the color grid, directly in the center of the dialog box.

To produce the 16 colors that you have available in HyperStudio, the six colors are mixed ("dithered") to produce new colors. The result of dithering is that you gain many new, additional colors over the original six. The disadvantage is that you can't directly edit one color; you must edit one of the original six, which in turn will affect various other dithered colors. In order to edit a dithered color, you must edit one or both of its components.

Each position of the color grid holds a color swatch which represents the two colors that were mixed to produce the color directly below it. For example, if you look at the second row from the top and third color swatch from the left, you'll notice that it's orange. Directly above the orange area is two smaller color swatches: red and yellow. These are the two "true" colors that were mixed to produce the dithered orange.

If you wanted to change this orange, you might want to change either the yellow component (represented by the third color swatch on the outside top edge of the grid) or the red component (represented by the second color swatch on the outside left side of the grid).

To edit a color, click the swatch on the outside of the grid that you wish to edit. The three components of the color (red, green, and blue) will be displayed in the editor window, at the bottom center of the dialog box.

The values for each component will be shown in numerical form to the left of the editor window, and each component will have a "thermometer" showing how much of it is used. For example, clicking the yellow swatch across the top of the grid shows that it has a value of 15 (the maximum possible) for its red and green components, and 0 (the minimum possible) for its blue component.

If you wanted to change the yellow to a green, you could click on the left arrow next to the red component, which would remove red from the yellow color and produce a more green color. Continuing to click the left arrow, and lowering the red component value, will produce a brighter green. (If you click 15 times, you'll bring the red value completely down to zero, leaving a solid green.)

If you decide that you want to revert back to pure yellow, a fast way to do that is to click the swatch again on the top of the grid, then click the yellow swatch to the right of the editor window (and the left of the "Done" button). This will copy that "pure" color back into the grid.

If you create a color table that you like and want to save it to disk, simply click the "Save Color Table" button. You will be prompted for a filename; the color table will be saved as a standard PaintWorks palette file.

To re-load your color table, click "Load Color Table." If you make changes to your color table and decide you don't want to save them, clicking "Cancel" will exit the color editor and discard your changes; otherwise, click "Done" to save them and exit.

Menu Tamer

by Michael O'Keefe

This extra will hide or show the menu bar on all cards in the current stack. After selecting "Menu Tamer," you will be asked if you want to hide all of the menu bars or show all of them. Note that Menu Tamer will never hide the menu bar on the current card.

Icon Maker

by Mike Nuzzi, Triad Venture

With this convenient extra, you can quickly and easily make icons from the image on the screen. These icons can be used with HyperStudio buttons, or in the Finder as regular icons.

To use Icon Maker, simply select "Make Icon" from the Extras menu. You will be presented with a short title and information dialog box, where you should click "OK" to continue (or "Cancel" to back up).

Next, the cursor will change to a crosshair, where you may draw a box (using the selector tool) around the object that you wish to change to an icon. With this crosshair, you are drawing a rectangle and the interface is similar to the HyperStudio Selector tool.

✔ **Advanced User**

You will only see the title and information dialog box one time. After that, the cursor will change to a crosshair immediately after you select "Make Icon" from the Extras menu.

After selecting the area of the screen that you want to be converted to an icon, you should click outside of the area. You will be prompted for a filename; the file will be saved as a normal Finder icon file with a filetype and auxtype of a Finder icon file and an application name of "HyperStudio". This means that you can also use Icon Maker to create icons that you can use in the Finder.

Note that whatever color is the background color will be transparent in the final icon—that is, if you have a picture of a brown donut on a blue background, the hole in the middle of the donut will be transparent in the icon that Icon Maker creates. The one exception to this rule is white, which is *always* opaque inside of an icon. (If you wish to make the white inside of an icon transparent, use the "Replace Colors" command, to change the white to yellow, or some other color which is otherwise not used in the icon. Then, change the background around the icon to the same color and then use Icon Maker. HyperStudio will see the yellow background and the yellow inside of the icon, and make the yellow inside of the icon transparent.)

Once you've made an icon, you can use it in HyperStudio to make an icon button. To convert an icon back into a screen image that you can edit, follow these steps:

1. Create an invisible button with the icon in question.
2. Save the screen to disk with the "Save screen..." command, under the File menu.
3. Delete the button.
4. Load the screen you just saved with the "Load background..." command, also under the File menu.
5. Edit the icon as you like with the paint tools.
6. Use Icon Maker again to change it back into an icon.

Box Maker

by Michael O'Keefe

This small extra is designed simply to show off how an Extra can draw with the paint tools. It allows you to quickly and easily draw three-dimensional boxes on the screen in the current color and line size.

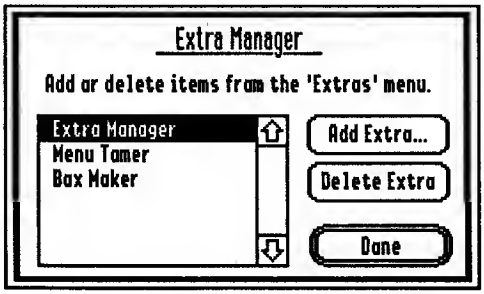
After selecting "Box Maker" from the Extras menu, the cursor will change to a crosshair and you may draw a rectangle as you would with the regular rectangle tool.

Once your box is drawn, Box Maker will "rubberband" a second rectangle, the same size as the one you had just drawn, with lines connecting the corners of both rectangles.

As you move the cursor around, the result will be a three-dimensional box. Click the mouse to complete the drawing process.

Extra Manager

by Michael O'Keefe



With the Extra Manager, you can add new Extras to HyperStudio and remove them, as well. Selecting "Extra Manager" brings up a dialog box with a list of all of the Extras you currently have in HyperStudio, and three buttons: "Add Extra...," "Delete Extra," and "Done."

Clicking "Add Extra..." will bring up a standard file dialog, prompting you for the disk file containing the Extra you wish to add to the Extras menu.

Clicking "Delete Extra" will remove the Extra you have selected from the list.

Finally, clicking "Done" will close the Extra Manager.

Transitions

The transitions are, for the most part, self-explanatory. The one exception is:

Half and Half

This transition is included on the disk as an example of a transition stored in a disk library. To use it, click "Disk library..." from the Transitions dialog, then open the `Half.and.Half` file.

Developer Opportunities

Transitions, Extras, New Button Actions (NBAs), and programming languages are all modular in HyperStudio—they are loaded from disk as needed and more of them may be added to the HyperStudio environment at any time.

For $10, Roger Wagner Publishing will be happy to send you our "Programmer's Resource Disk" for writing HyperStudio add-ons. It will explain how to write each of the items mentioned above, and includes complete information on interfacing with HyperStudio as well as many source code examples.

If you can't program, don't be discouraged! Developing stacks for Roger Wagner Publishing's *StudioWare Catalog* is a way to turn your HyperStudio hobby into hard cash. If you have an idea for a stack, or you already have a stack which you'd like to sell through our catalog, please send it to Steve Allen, Director of Software Development, at Roger Wagner Publishing. We're always interested in looking at new submissions and would love to see your creations.

Finally, if your interest isn't in stack building, but rather a component of stack design (such as drawing artwork or digitizing sound effects), we'd like to talk to you. This might be just the case if you're a computer artist, for example—there's always a need for quality artwork in a hypermedia development system like HyperStudio, and we can help you get in touch with stack designers looking for your services.

Glossary

Here, many of the terms used in this manual are defined.

A, B

A is for Apple. The Apple key is labeled on most keyboards with an apple symbol. It is also called the Command key.

area button: Three area buttons are available in HyperStudio:

Freehand Area Button. This type of button is simply drawn with a tool similar to the Pencil tool.

Expanding Area Button. This type of button "expands" to fill the inside of the shape that you specify.

Lasso Area Button. This type of button contracts around a shape you specify, hugging the outer border of the shape.

ASCII: Abbreviation for American Standard Code for Information Exchange. The ASCII system assigns a number to each character that a computer can produce. For example, ASCII character 65 is the letter "A".

auto-activate: An auto-activate button presses itself after a certain length of time. It could "go off" when a user goes to a card, or after a time delay that you specify.

back list: A list (maintained by the computer) of the last sixteen cards you visited.

background: a screen that can be shared by many cards. Objects (such as buttons) are drawn on top of the background. The paint tools modify the background.

Browse tool:  This tool lets you click buttons and look through stacks. Clicking the Browse tool inside of an editable text item allows you to type in the text item.

byte: a unit of measurement for the RAM in your computer. One byte represents one character.

C, D

clear: This command erases whatever is selected—an object or a range of text. Note that the selected item (or range of text) is not placed on the clipboard, and any item already on the clipboard will not be replaced.

click: Press and release the mouse button. Clicking an object means to move the mouse pointer so that it points to the object, then pressing and releasing the mouse button.

clipboard: A holding place for items that have been cut or copied. Only one item can be on the clipboard at a time.

Command key: also called the Apple key. On most keyboards, this key is labeled with an Apple symbol, the "propellor" symbol, or the word "command."

copy: This puts a copy of what was selected onto the clipboard. The original will stay in place. See **Paste.**

current card: the card currently being displayed on the screen.

cursor: see **insertation point.**

cut: Cutting an object (or range of text) will remove it from the stack and place it on the clipboard. See **Paste.**

default: This means the option that is already chosen or "preset at the factory." For example, as a default, a button does not play a sound without you specifically marking "Play a sound..."

deselect: A selected item is highlighted or has a frame of moving dots around it. Clicking elsewhere on the screen, away from the selected item, will *deselect* it and remove the highlighting.

dialog box: A box on the screen that requires you to click somewhere or enter information. Throughout HyperStudio, selecting options that end with ellipses ("...") will open a dialog box asking you for more information.

digitize: To convert something into information recognized by the computer. Recording a sound or scanning a picture are ways of digitizing information.

digitizer card: The circuit board included with HyperStudio is a digitizer card. It converts sounds from a microphone or other source into data that HyperStudio can use.

dimmed: Menu and dialog selections might not be valid in some situations—they are shown *dimmed.* The letters are gray instead of black.

directory: See **folder.**

disk-based data: A disk-based sound or word processing file is stored on disk and is brought into memory from disk only when HyperStudio needs the information it contains. The actual sound or text is not stored in the stack. This is also called extended data.

dither: to mix two colors to produce a third. HyperStudio's 16 colors are actually produced by mixing together ("dithering") six colors in different combinations.

drag: Dragging an object allows you to move it around on the screen. To drag an object, you move the mouse pointer to the object, and then hold the mouse button down. With the mouse button held down, move the mouse, and the object will move with the mouse pointer. When you have the object in the location where you want it, release the mouse button to "drop" the object.

E, F, G

editable text: This is text in a text item that can be changed (edited) by the user. The opposite of **fixed text.**

embedded: Items that are embedded are stored as part of the stack. The opposite of **disk-based data.**

extended commands (XCMDs): see **New Button Actions (NBAs).**

extended data: see **disk-based data.**

field: This another word for a text item.

fixed text: This text cannot be changed by a user. Opposite of **editable text.**

folder: Files can be stored together in a folder. When you save data from within HyperStudio, you have an option to create a new folder for information. Also known as a **subdirectory.**

grayed: see **dimmed.**

group: Objects and cards can be part of a group. In that case, the objects and the card's background is shared with all of the other members in the group. Changing a grouped item changes it on all cards in the group.

GS/OS: The Apple IIGS' operating system. You must have the version that comes on **System Disk** 5.0.4 or newer.

H, I, J

handle: small squares on the edge of a rectangle, that allow you to change the size of the rectangle by dragging.

hard link: a link from a button to another card, where the destination card number is stored in the button definition. See **soft link.**

highlight: Objects are highlighted to draw attention to them. Frequently, after selecting an object, it will become highlighted. Highlighting is done by reversing the colors of the object (when selecting a range of text), or by coloring the outline (such as when selecting a button or graphic selection).

home card: The first card of the home stack. Used interchangeably with **home stack.**

home stack: The central navigating point for HyperStudio. HyperStudio tries to load the file named `Home.Stack` when it first starts; frequently, a home stack will contain nothing but buttons leading to other programs and stacks.

I-beam: This is the shape the pointer takes in an area for entering text. If you click the mouse while the pointer is an I-beam, it places the cursor for you to type text.

icon: a small picture. With HyperStudio, icons may be added to buttons (making them "icon buttons"). The Apple IIGS Finder uses icons to represent files and folders stored on a disk.

insertion point: A blinking vertical bar, indicating that you may type to insert text. You can change the location of the insertation point when the cursor is an **I-beam.**

K, L, M

key color: This is the "show-through" color when using the Video Overlay Card and a laserdisc player. Anything in this color will appear transparent and the video will show through it.

keyboard equivalent: A combination of keys that perform the same action as a menu selection. For example, pressing and holding the Apple key, then tapping the H key (abbreviated as Ó-H) takes you to the home stack.

kilobytes: a measurement of the amount of RAM in your computer, abbreviated with the letter K. One kilobyte is one thousand bytes (one thousand characters). See also **megabyte.**

Klimas, Dave: one of the primary authors of HyperStudio. According to Dave, HyperStudio 3.0 "sounds good to me."

last-used directories: HyperStudio remembers where you keep certain types of data, and remembers the last directory that you accessed to get to that data. For example, if you choose "Add Clip-Art..." to put some art in a stack, and then use "Load Background," HyperStudio will remember the folder where you got the art from (when you added clip-art) and automatically bring you to that directory to load a background graphic.

launch: To start another program. For example, a button in a stack may *launch* AppleWorks, which means that it causes HyperStudio to shut down and AppleWorks to startup.

marked card: a card that has its "marked" attributed set (in the "Features..." dialog box). Buttons may be set to return to the most recent marked card the user visited.

marker bar: A place marker in a waveform, in Sound Shop. Clicking once in the waveform places the marker bar, which is used to indicate the place that certain editing functions (such as "Paste") will occur.

megabytes: One thousand kilobytes (actually 1,024K). The standard memory configuration for an Apple IIGS is 1.25 megabytes.

menu bar: The bar across the top of the screen that lists the menu titles ("File," "Edit," and so on).

modifier: the Apple (Command), Shift, Control, or Option key. These are called the "modifier keys" because they *modify* the action of another key. (Pressing the "V" key alone is different from pressing Command-V, or Option-V.)

Mueller, Eric: one of the primary authors of HyperStudio. Eric's hobbies including revising HyperStudio reference manuals and enjoying compact discs (especially those by OMD, Jean-Michel Jarre, the B-52's, and Erasure). He likes to spend his weekends drinking gallons of Diet Coke in West Hollywood and at Disneyland with his faaaaaabulous friends Dave, John, and Terry.

N, O, P

New Button Actions (NBAs): Accessed from the Button Actions screen, NBAs are programs that extend the functionality of HyperStudio. For example, an NBA may change the border color or dial the telephone. HyperStudio has a built-in library of NBAs, and the capability to load new NBAs from disk.

O'Keefe, Michael: one of the primary authors of HyperStudio. In his own words (sort of), he says, "Hi, my name is Michael, and I'm a Pisces. I love computers, and hot tamales!" Michael also wants to be king of the universe.

object: buttons, text items, and graphic objects are all objects. They all can be part of a group, may be edited, and float above the card's background.

painted text: Text painted with the text tool. This text may only be edited with the other paint tools.

palette: another name for a color table. HyperStudio uses a **dithered** palette of 16 colors.

Paste: To move a copy of the data from the clipboard to the current card. Objects, parts of the background, and even entire cards can be pasted. See **copy** and **cut.**

pixel: One dot on the screen is a pixel. Pixel is short for "picture element."

playback rate: The rate at which the computer plays a sound. Generally, this number is set to the same value as the **record rate** of the sound.

Q, R, S

RAM: An acronym for Random Access Memory, this refers to the amount of memory your computer has. RAM is measured in **kilobytes** and **megabytes.**

record rate: The rate at which the computer records a sound. This number represents how many times the computer is storing information *per second.* The higher the number, the better the quality of recording, and the more memory required for the sound.

regional button: see **area button.**

resource fork: The resource fork is a special location in a stack where sounds, icons, NBAs and transitions attached to a button are stored. Once they are in the resource fork, other buttons may use them without having to place a second copy in the resource fork.

run-time: a special version of HyperStudio, named `HS.Sys16`, that you may give away with your stacks.

scroll: To move through the contents of a window or text item.

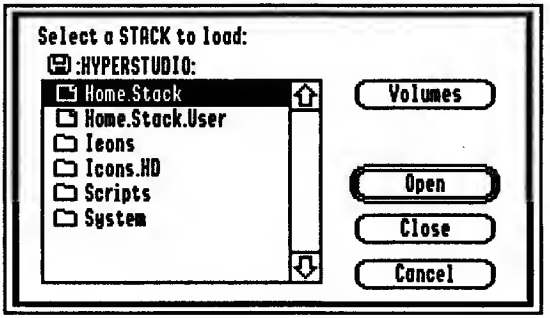
SimpleScript: The programming language built into HyperStudio. SimpleScript is very similar to English.

Smith, Jeff: one of the primary authors of HyperStudio. Jeff's hobbies include talking with a silly accent, juggling, and role-playing. Jeff has been known to hide under the pseudonym of "Jeff Halsey" so that people can't find him.

soft link: a link from a button to another card, where the destination card is *not* stored as an absolute card number but instead just an offset. See **hard link.**

sound input device: Whatever is connected to the HyperStudio digitizer card is a sound input device—this could be the microphone, or a cable to a tape player.

standard file dialog: This is used to load or save files. Click "Volumes" to look in other drives.



standard file dialog

subdirectory: A folder within a folder.

System Disk: the System Disk comes with every Apple IIGS computer. Included on the System Disk are **GS/OS,** the computer's operating system, and the Finder, a disk utility program.

T, U, V

target: The destination. For example, referring to the "target card" of a button usually refers to which card the user will be placed on when they click the button.

text field: see **text item.**

text item: One of the three main objects in HyperStudio. Text typed directly into the text item and stored as part of the stack is *embedded* text. Text stored in a file on the disk (instead of being part of the stack) is *extended* data. A text item can be *editable* (meaning the user can type or change text in the item) or *fixed* (read-only).

text tool: This tool lets you *paint* text. Painted text becomes part of the background, and is not part of a text item. It can be edited only by being repainted.

toggle: To switch between two choices. For example, you can toggle between a button playing a sound or not playing a sound by simply marking and unmarking the "Play a sound..." checkbox in the Button Actions dialog box.

transition: This is the visual effect that happens between cards. HyperStudio comes with a number of transitions built into the program, and has the capability to add more.

undo: A command that undoes the last editing or formatting command. Not all actions can be undone.

unresolved button: A button that no longer has a destination. This could happen if you create a button that points to a specific card in a stack, and then you delete that card.

user disks: disks which contain your stack(s) and the file `HS.Sys16`, the run-time version of HyperStudio. A user disk is what you would give to someone who doesn't own HyperStudio but wants to look at your stacks.

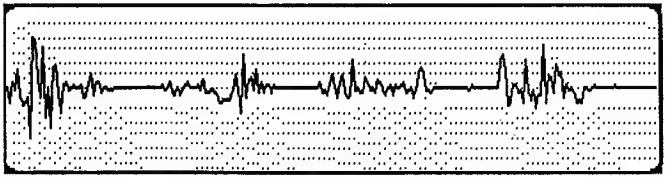
Video Overlay Card: This piece of hardware allows you to show video from a laserdisc player (or other video source) on the Apple IIGS monitor and mix computer graphics with the video. HyperStudio controls the Video Overlay Card.

VOC: see **Video Overlay Card.**

W, X, Y, Z

Wagner, Roger: the primary designer of HyperStudio. A brilliant mind with strange neckties, and vice versa.

waveform: The picture of a sound, as displayed in Sound Shop.



a waveform

wrap-around effect: Cards in a stack are arranged in an imaginary ring; selecting "Next card" from the last card in a stack brings you to the first card. This is called the "wrap-around" effect because you wrap back around to the first card after the last card. Similarly, this also applies when moving from the first card to the "Previous card" (which takes you to the last card in the stack).

XCMDs: see **New Button Actions (NBAs).**

Index